Fault Tolerant Flight Control System Design for Unmanned Aerial Vehicles

A thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

Rudaba Khan

Bachelor of Engineering (Aerospace), RMIT University

Bachelor of Applied Science (Aviation), RMIT University

Bachelor of Applied Science (Mathematical Sciences), RMIT University

School of Mathematical and Geospatial Sciences

School of Aerospace, Mechanical and Manufacturing Engineering

College of Science Engineering and Health

RMIT University

Australia.

August 2016

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; any editorial work, paid or unpaid, carried out by a third party is acknowledged; and, ethics procedures and guidelines have been followed.

Rudaba Khan

July 4, 2016.

Acknowledgements

First and foremost my sincere thanks to Dr. Paul Williams without whom this research would not exist. Thank you Paul for teaching me and helping me understand MPC, for helping me with developing ideas and the various Matlab models, for assisting me in putting the initial thesis structure together and for providing me with the UAV aircraft model. Thank you for taking time out of your busy schedule and for putting up with me over the years and all my annoying questions, for being patient with me and for sharing your knowledge and expertise from which I have benefited a great deal. I immensely appreciate all that you have taught me. You have not only been an excellent teacher and mentor but also a great friend so thank you.

To my senior supervisor A/Prof. Asha Rao, my "Juggernaut", how will I ever repay you? You saw me floundering, threw me a lifeline and pulled me out of a very deep hole. I cannot begin to put into words how appreciative I am. You taught me to believe in myself again, your words of wisdom and encouragement helped me stand up straight and you always knew what to say at the right times and make me feel better again. Thanks largely to your eagle eyes, hard work, dedication and sound judgement regarding the write up of this thesis that I am very proud of the final outcome. You care deeply for your students and have an incredible work ethic which I sincerely hope to mimic one day. I stand here today because of you and I am extremely honoured to have you as a part of my life, a loving friend and great mentor.

To my co-senior supervisor Dr. Robin Hill, thank you for sticking by me throughout all these years it has been a long journey to say the least but we got there. Thank you for all of your help and discussions in understanding difficult concepts and for supporting me during my tough times. Thank you also for all of your hard work and dedication I could not have finished without your help. It has been a pleasure working with you, thank you for being a mentor as well as a supportive friend.

Thanks to my dear friend / 2^{nd} mother Carmel, who encouraged and supported me and listened to me for hours on end and helped me through one of the most difficult times in my life. You helped me stop and think about why I loved my work so much and why I pursued my PhD in the first place. Your tough love (although harsh at times) was exactly what I needed and I love you for it. You also forced me to realise that I should never be embarrassed or ashamed to ask for help and you brought Paul Riseborough back into my life.

Which leads me to Dr. Paul Riseborough. Thank you so very much for all that you have done for me, again I can not put into words how grateful I am. You are an incredibly busy person yet you still found time to help me, you took me on as a student again and became apart of my supervisory team for which I am forever indebted because without you I would not have reached the finish line. You are held in high esteem by everyone in the aerospace community and I am so honoured to have had the privilege to have worked with you and most of all to have been taught by you. Thank you in particular for all of your help with the filter tuning, PID control development, for re-focusing the research and for your assistance with the UAV case study. I am humbled to have worked with you, Dr. Paul Williams and Dr. Michael Crump. The fondest memories of my career thus far are those that I spent learning from you guys at BAE SYSTEMS. You three were and are my inspirations for wanting to pursue a career in control systems engineering and I have always believed and still do that if I can be one tenth as good a engineer as any one of you I know I will make a great engineer. I have always been in awe of your passion and knowledge for aerospace. Not only that, but you take pride and pleasure in imparting your knowledge and experience to younger engineers which is a true gift. I greatly appreciate the time you have spent with me on this thesis I have learned an enormous amount from you and hope to continue to learn from you. Thank you too for being an excellent mentor and an even more fantastic friend.

A special thank you also to my second supervisor Associate Professor Cees Bil. You have been a prominent figure in my undergraduate career as well as in my postgraduate career for which I am very appreciative. I still remember the day I approached you to talk about taking on a PhD. all those years ago, you were incredibly helpful. You were the one who suggested this topic, thank you so very much I am so pleased that I found you that day at RMIT, I have immensely enjoyed working in the area of fault tolerant flight control design. Thank you for all of your support over the years.

Thanks to my PhD friends, Ummul Fahri Abdul Rauf, Reza Roozbahani, Jessica Dunn, Solmaz Jihad, David Ellison and Jessica Liebeg for keeping it real for me and making me laugh and for all of your help and support. Thank you in particular to Solmaz, David, Jessica L. and Isaac for all the tea times I always looked forward to them everyday. Thank you also to my lucky charm My-Anne Nguyen, our chat before my completion seminar really helped settle my nerves

and it ended up being a huge success, thank you for being such a good friend. The PhD journey can often be quite lonely hence I'm grateful to have met some fantastic people from all parts of the world, I have made true friends for life. A special thanks to David Ellison for his invaluable discussions on the Brachistochrone problem and parts of the 2D robot model. Thanks to my best friends Kylie Brickley and Amber Buse my cousin Sidra Khan, my good friend Uzma Chohan (Uzma Aunty), Appah (Valli) and my Khalto Lina for your continuous support and encouragement and for always (without any hesitation) lending me your shoulders to cry on.

Thank you also to all the staff at the SMGS department at RMIT University I think of you all as my family. Thank you for providing a supportive and encouraging research environment it truly makes coming into University a joy.

A huge thank you to my big brother Shahid bhai. You really pushed me to finish my PhD, you have been a massive support for my family and I, in particular for my father for which I am very grateful. Thank you for the Dua (prayer) reciting it was honestly a turning point in my PhD for me.

Last but definitely not least to my family, my brothers Omar and Abid, my sister-in-law Salma, my niece Alisha and most of all my parents. Salma you are the sister I never had, I could not have asked for a better sister you have gone above and beyond in your support for me. I do not know how I would have coped without you especially during these last 2 years, so thank you from the bottom of my heart. My brothers Omar and Abid (my Chehnna), both of whom are also believers in tough love, I could not imagine my life without you two and I take great pride in the fact that you have both stood by me. Thanks especially to Chehnna for listening to my dribble I know I annoy you at times but I also know you are always there to listen, you are an awesome little brother and I could not ask for a better brother, you are an extraordinary young man, thank you for always having my back. Thank you to my little girl Alisha for bringing sunshine to my life. To my parents I love you both so much, you have always encouraged me to be my best and to stand on my own two feet. Thank you for teaching me to be strong and independent and thank you for always believing in me and being by my side. I stand here today because of the two of you, because of your love, support and encouragement. I know that no matter what, good or bad, you two will always be there for me and I for you, I am so blessed to have such wonderful parents.

This PhD. journey has been a long and bumpy ride. The main thing I have learned from this experience is how lucky I am to have such an amazing support network. I am truly grateful for all the wonderful people in my life. Thank you to all of my friends and family who have helped support me through out this journey.

Contents

List of Figures

List of Tables

Acronyms

AC	adaptive control.
AHV	air breathing hypersonic vehicle.
AI	artificial intelligence.
ANN	artificial neural network.
AoA	Angle of Attack.
AR	analytical redundancy.
ARE	algebraic Riccati equation.
BVP	boundary value problem.
c.g.	centre of gravity.
CA	control allocation.
CGL	Chebyshev-Gauss-Lobatto.
CPU	central processing unit.
DAE	differential algebraic equation.
DCM	direction cosine matrix.
DE	differential equation.
dem	demand.
DI	dynamic inversion.
DMC	dynamic matrix control.
DoF	degrees of freedom.
EA	eigenstructure assignment.
EAS	equivalent airspeed.
EKF	extended Kalman filter.
EMMAE	extended multiple model adaptive estimation.
EPSAC	extended prediction self adaptive control.
FCC	flight control computer.

FCS	flight control system.
FDD	fault detection and diagnosis.
FDI	fault detection and identification.
FDM	finite difference method.
FEM	finite element methods.
FL	feedback linearisation.
FTC	fault tolerant control.
GIMC	generalised internal model.
GL	Gauss-Lobatto.
GNC	guidance, navigation and control.
GPC	generalised predictive control.
GS	gain scheduling.
IC	intelligent control.
IMM	interactive multiple model.
IVP	initial value problem.
KKT	Karush Kuhn Tucker.
LGL	Legendre Gauss Lobatto.
LMF	linear model following.
LMI	linear matrix inequality.
LoC	Loss of Control.
LPV	linear parameter varying.
LQR	linear quadratic regulator.
LTI	linear time invariant.
MAC	model algorithmic control.
meas	measured.
MF	model following.
MFRFC	model following reconfigurable flight control.
MIMO	multiple input multiple output.
MISO	multiple input single output.
MM	multiple model.

MMAE	multiple model adaptive estimation.
MMST	multiple model switching and tuning.
MPBVP	multi-point boundary value problem.
MPC	model predictive control.
MPHC	model predictive heuristic control.
MPIM	modified pseudo inverse method.
NASA	National Aeronautics and Space Administration.
NED	North, East, Down.
NLP	nonlinear programming problem.
NMPC	nonlinear model predictive control.
NN	neural network.
ODE	ordinary differential equation.
PCA	propulsion controlled aircraft.
PDE	partial differential equation.
PID	proportional integral derivative.
PIM	pseudo inverse method.
PMF	perfect model following.
PMP	Pontryagin's maximum principle.
QFT	quantitative feedback theory.
RoC	rate of climb.
SISO	single input single ouput.
SMC	sliding mode control.
SQP	sequential programming problem.
TAS	true airspeed.
UAV	unmanned aerial vehicle.
UKF	unscented Kalman filter.
US	Unites States of America.
VSC	variable structure control.

Nomenclature

α	Angle of attack
$\bar{c}$	Mean aerodynamic chord
$\bar{q}$	Dynamic pressure
β	Angle of sideslip
Δt	Timestep
δ	Perturbation
δ_a	Aileron deflection
δ_e	Elevator deflection
δ_r	Rudder deflection
δ_{th}	Throttle
$\hat{\mathbf{z}}$	Predicted measurement
κ	UKF scaling factor
Λ	Probability likelihood
$\dot{\mathbf{x}}$	State derivative
ν	Innovation
ω_{wind}	Wind turbulence
$\mathbf{F}$	State transition Jacobian matrix
$\mathbf{H}$	Measurement Jacobian matrix

$\mathbf{K}$	Kalman gain
$\mathbf{P}$	State covariance matrix
$\mathbf{Q}$	Process noise matrix
$\mathbf{R}$	Noise covariance matrix
$\mathbf{S}$	Innovation covariance
$\mathbf{u}$	Control input vector
$\mathbf{u}_{lb}$	Lower bound control input constraint
$\mathbf{u}_{ub}$	Upper bound control input constraint
$\mathbf{V}_{\text{wind}}$	Wind Velocity
$\mathbf{v}$	State noise
$\mathbf{w}$	Measurement noise
$\mathbf{x}$	State vector
$\mathbf{x}_{lb}$	Lower bound state constraint
$\mathbf{x}_{ub}$	Upper bound state constraint
$\mathbf{z}$	Measurement vector
$\mathcal{L}$	Bolza component of cost function
$\mathcal{M}$	Mayer component of cost function
J	Advance ratio
μ	IMM mixing probability
ω_L	Left wheel angular rate
ω_R	Right wheel angular rate
ω_{prop}	Propeller angular speed
ϕ	Roll angle
ψ	Heading angle

ψ	Heading angle
ψ_0	Initial point conditions
ψ_f	Final point conditions
ρ	Air density
CT	Non-dimensional thrust coefficient
θ	Pitch angle
A	System State Matrix
a_D	Accleration in navigation frame, down-direction
a_E	Acceleration in navigation frame, east-direction
a_N	Acceleration in navigation frame, north-direction
a_X	Acceleration in body axis, x-direction
a_Y	Acceleration in body axis, y-direction
a_Z	Acceleration in body axis, z-direction
B	System Input Matrix
b	Distance from C to centre of wheel
b	Wing span
C	Geometric centre of mobile robot
C_b^n	Body to navigation frame direction cosine matrix
C_l	Non-dimensional rolling moment coefficient
C_m	Non-dimensional pitching moment coefficient
C_n	Non-dimensional yawing moment coefficient
C_n^b	Navigation to body frame direction cosine matrix
C_X	Non-dimensional force coefficient in body axis, x-direction
C_Y	Non-dimensional force coefficient in body axis, y-direction

C_Z Non-dimensional force coefficient in body axis, z-direction

$D_{j,k}$ Differentiation matrix

g Gravity

g_L Lower bound path constraints

g_U Upper bound path constraints

H_p Prediction horizon

H_u Control horizon

I_p Integral error in roll

I_q Integral error in pitch

I_r Integral error in yaw

J Cost function

k Current time

K_D Derivative gain

K_I Integral gain

K_P Propotional gain

M Number of sub-intervals

N Number of coincidence points

N_u Control points

n_u Number of input states

N_x State points

n_x Number of states

p Roll rate

q Pitch rate

Q_ψ MPC wheel angular rate weighting matrix

Q_I MPC integral error weighting matrix

Q_p MPC aircraft angular rate weighting matrix

Q_u MPC control input weighting matrix

Q_V MPC velocity weighting matrix

Q_x MPC state weighting matrix

$Q_{\dot{\psi}}$ MPC yaw rate weighting matrix

R Radius of robot wheel

r Yaw rate

R_L Left wheel radius

R_R Right wheel radius

S Wing area

T Thrust

t Time

t_0 Initial time

t_f Final time

t_s Sample time

u Velocity in body frame, x-direction

u_0 Nominal input value

v Velocity in body frame, y-direction

V_D Down velocity

V_E East velocity

V_N North velocity

v_s Speed of sound

V_T True airspeed

V_{stall}	Stall speed
w	Velocity in body frame, z-direction
w_j	LGL weights
x	x-coordinate
x	y-coordinate
x_0	Nominal state value
x_D	Down direction coordinate
x_E	East direction coordinate
x_f	Final x-coordinate
x_N	North direction coordinate
y	y-coordinate
V	Speed

Summary

Safety and reliability of air vehicles is of the utmost importance. This is particularly true for large civil transport aircraft where a large number of human lives depend on safety critical design. With the increase in the use of autonomous unmanned aerial vehicles (UAVs) in our airspace it is essential that UAV safety is also given attention to prevent devastating failures which could ultimately lead to loss of human lives. While civil aircraft have human operators, the pilot, to counteract any unforeseen faults, autonomous UAVs are only as good as the on board flight computer. Large civil aircraft also have the luxury of weight hence redundant actuators (control surfaces) can be installed and in the event of a faulty set of actuators the redundant actuators can be brought into action to negate the effects of any faults. Again weight is a luxury that UAVs do not have. The main objective of this research is to study the design of a fault tolerant flight controller that can exploit the mathematical redundancies in the flight dynamic equations as opposed to adding hardware redundancies that would result in significant weight increase. This thesis presents new research into fault tolerant control for flight vehicles.

Upon examining the flight dynamic equations it can be seen, for example, that an aileron, which is primarily used to perform a roll manoeuvre, can be used to execute a limited pitch moment. Hence a control method is required that moves away from the traditional fixed structure model where control surface roles are clearly defined. For this reason, in this thesis, I have chosen to study the application of model predictive control (MPC) to fault tolerant control systems. MPC is a model based method where a model of the plant forms an integral part of the controller. An optimisation is performed based on model estimations of the plant and the inputs are chosen via an optimisation process.

Linear model predictive control (LMPC) has been studied for more than three decades and is a well established control methodology commonly used in the process industry but is only now making headway into industries such as aerospace. Nonlinear model predictive control (NMPC) on the other hand has not received much attention due to limitations in computing power, but with the advancements in computing power the application of nonlinear model predictive control (NMPC) is now becoming more viable and it forms the main focus of my research. One of the main contributions of this thesis is the development of a non-linear model predictive controller for fault tolerant flight control. An aircraft is a highly non-linear system hence if a nonlinear model can be integrated into the control process the cross-coupling effects of the control surface

contributions can be easily exploited. Another reason for choosing nonlinear model predictive control over linear model predictive control is that the linear method requires small perturbations to work efficiently and effectively. However, if a fault occurs it is likely to take the system out of the linear region where the perturbations from the nominal are quite large, hence the underlying assumptions would be violated causing the solution to diverge. Nonlinear techniques, on the other hand, are better able to handle these situations as they encompass the whole system operating envelope.

Non-linear MPC requires the solution of an optimal control problem which is most commonly solved via a direct solution method using a finite parameterisation of the control and/or constraints. The most commonly used parameterisation methods are single direct shooting and direct multiple shooting. The method proposed in this thesis uses a Pseudospectral discretisation method. It is shown through an illustrative example that the Pseudospectral discretisation method used within an NMPC framework can achieve the same level of accuracy as the most commonly used methods using fewer number of discretisation points.

Finally an active fault tolerant control system comprises not only of the fault tolerant controller but also a fault detection and isolation subsystem. A common fault detection method is based on parameter estimation using filtering techniques with the most commonly used filter being the extended Kalman filter (EKF). The solution proposed in this thesis uses an unscented Kalman filter (UKF) for parameter estimation and controller updates.

In summary the main contribution of this thesis is the development of a new active fault tolerant flight control system. This new innovative controller exploits the idea of analytical redundancy as opposed to hardware redundancy. It comprises of a non-linear model predictive based controller using pseudospectral discretisation to solve the nonlinear optimal control problem. Furthermore a UKF is incorporated into the design of the active fault tolerant flight control system. The filter provides fault estimations and model parameter updates to the controller.

Chapter 1

Introduction

1.1 Motivation

The increase in cost and complexity of autonomous unmanned aerial vehicles (UAVs) has resulted in a shift in the design philosophy away from expendable systems towards systems with a high level of robustness and survivability. Traditionally, survivability is increased through redundancy of critical systems, but this increases weight, complexity, and cost. Although system redundancy is inevitable in developing a commercial product, most systems are not robust beyond first failure. Some other means needs to be found that will allow the vehicle to navigate to a specified point and land safely after a system or actuator failure.

The overall objective of this research is to develop and implement an innovative flight control system applicable to UAVs, whereby the system is capable of reconfiguring the controller to adapt to identified faults in the system. In so doing, the reconfigured controller is able to recover adequate authority on the aircraft and automatically bring it back to the ground in a safe and controlled manner, thus avoiding a crash or catastrophic loss of equipment. More specifically the aims are to:

1. Develop autonomous UAV control technologies that have the capability to reconfigure their structures and adapt to a new operational environment in the case of fault occurrence, while respecting system operational constraints.

2. Apply model predictive control (MPC) technologies to a UAV test bench by model-in-the-loop simulation.

1.2 Outline

This chapter puts into context the scope of this research and briefly outlines its importance. The main application of this research is the development and improvement of UAV technologies hence it is important to understand what a UAV is, therefore section 1.3 concisely defines unmanned aerial vehicles. This is followed by details of the contributions of this thesis towards the advancement of UAV technologies in section 1.4.

Finally an overview of the thesis is given in section 1.5.

1.3 Unmanned Aerial Vehicles (UAVs)

By definition UAVs are aerial vehicles capable of sustained fight without a human operator on board [63]. Many UAVs are remotely piloted. However, to be truly autonomous, sophisticated control systems need to be integrated into the UAV to enable the vehicle to fly without any human intervention.

An important feature for true autonomy is the ability of UAVs to *"think on their own"*. This includes the ability to respond quickly to system failures. UAVs, unfortunately, are currently deemed to be unsafe due to the fact that they are unreliable [31], mainly because the UAV crash rate is higher than that of manned aircraft. In [31] the leading cause of UAV accidents was found to be human error due to a lack of situational awareness by the remote pilot. In addition a number of UAV accidents are caused by component failures or operator error. Similarly Bateman [12] argues that poor reliability records, absence of certification standards and regulations with regards to UAV systems have hindered their integration into the civil airspace. At present, more emphasis is placed on reducing the unit cost per UAV rather on reliability, but if we are to increase trust in UAVs it is imperative that we increase reliability/predictability.

1.4 Thesis Contributions

The main contribution of this thesis is the development of a new active fault tolerant control system for UAVs comprising of a non-linear model predictive based controller using pseudospectral discretisation to solve the nonlinear optimal control problem along with an Unscented Kalman Filter for fault detection and identification. The controller has been demonstrated to work effectively on a 2D robot model, a generic aircraft model and an actual UAV model.

1.5 Thesis Overview

The thesis is divided into seven chapters. Each chapter begins with an introduction giving the motivation behind the contents of the chapter followed by a brief outline. The following is a synopsis of the chapters herein.

Chapter 1 - Introduction

Chapter 1, this chapter, places the research into context, summarising the thesis contributions and providing a brief overview of the thesis structure.

Chapter 2 - Review of Recent Advances in Fault Tolerant Flight Control

Chapter 2 provides a detailed description of the problem addressed in this thesis, namely fault tolerant control for UAVs and aims to explain in detail why this research is important. This chapter takes an in depth look at what has been achieved in the area of fault tolerant control thus far.

Chapter 3 - Design of a Nonlinear Model Predictive Controller

Model predictive control MPC is an advanced control methodology. The aim of Chapter 3 is to explore this technique for use as a fault tolerant controller. Both the theoretical and practical aspects of linear and nonlinear MPC are explored.

In this Chapter, a number of the numerical techniques used in the implementation of nonlinear model predictive control (NMPC) will be investigated, to gain a thorough understanding of their capabilities as well as their limitations. As a result of this analysis one method will be chosen to design both linear and nonlinear MPC controllers for a simple 2D robot model and performance results compared.

Initially the numerical techniques will be applied to the well known Brachistochrone problem, with the pseudospectral discretisation method turning out to be the most promising in terms of accuracy and savings in computational time. The pseudospectral method will then be used to design both linear and nonlinear MPC controllers. In general MPC solves an open loop problem at each time step hence initially the controllers will be set up for the 2D robot to solve the open loop problem and a sensitivity analysis conducted. From the analysis a prediction window length of 5 seconds along with 50 coincidence points are chosen to carry out the remainder of the

research. The chapter concludes with performance comparisons between linear and nonlinear MPC applied to the closed loop problem where the robot is required to follow a demanded trajectory. From the analysis, it will be evident that for small perturbations the performance of both linear and nonlinear MPC is similar however as the perturbations increase the performance of NMPC becomes far superior.

Chapter 4 - Fault Tolerant Control System Design

Fault detection and identification is an integral part of fault tolerant design. Chapter 4 is dedicated to investigating various fault detection techniques with the 2D robot model used again as an illustrative example to test the selected methods. The chapter concludes with a full fault tolerant control (FTC) system design.

The controller design of chapter 3 is integrated with a number of fault detection and identification (FDI) filtering techniques to develop a full active FTC system. As a result of the analysis the design of the FTC system to be implemented as a fault tolerant flight control system is finalised. This final design comprises a pseudosepectral NMPC based controller integrated with an unscented Kalman filter (UKF) filter for fault detection and identification.

Chapter 5 - Fault Tolerant Flight Control System Design

In Chapter 5 the fault tolerant controller designed in Chapter 4 is applied to flight control. A generic aircraft model is used to explore the feasibility of the proposed design as a fault tolerant flight controller. Parts of section 5.4 and 5.5 of this chapter have appeared as [72] and [73].

Initially the FTC system is applied to the longitudinal (or 3DoF) motion of the aircraft assuming FDI followed by a look at the full 6DoF motion of flight. Several problems were encountered during the design of the FDI subsystem for the 6DoF motion and further research in this area was identified as beyond the scope of the current research. Thus the research is redefined by narrowing the scope and demonstrating the concept of the proposed FTC system design on the longitudinal motion of the aircraft. The chapter concludes with the successful application of the proposed active FTC control system design to flight control.

Chapter 6 - UAV Case Study

The fault tolerant flight control system of Chapter 5 is applied to an actual UAV model in Chapter 6. In this chapter the feasibility of the application of the proposed design on a UAV is effectively demonstrated via model-in-the-loop simulation.

Chapter 7 - Conclusion and Recommendations for Further Work

The final chapter of the thesis summarises the main findings and contributions of this research. The chapter ends with recommendations for further work.

The next chapter (Chapter 2) presents a thorough literature survey on fault tolerant flight control.

Chapter 2

Review of Recent Advances in Fault Tolerant Flight Control

2.1 Introduction

2.1.1 Motivation

This thesis studies the design of fault tolerant flight control systems for unmanned aerial vehicles (UAVs) to prevent catastrophic failures. It is important to place this study in context and this chapter aims to do just that, looking at past research into fault tolerance, in particular, for flight control, at the various components of a fault tolerant control (FTC) system and the application of model predictive control (MPC) to fault tolerant flight control.

2.1.2 Outline

Beginning with a discussion on the role of UAVs and the need for fault tolerance (section 2.1.3), I move on to look at the history of flight control systems in section 2.2. This is followed by a review of the current state of research in fault tolerant control systems, and the control techniques currently used therein (section 2.3). Fault detection and identification, an important part of FTC, is touched on briefly in section 2.4 as a detailed description is provided in the introduction of chapter 4. Section 2.5 presents the state of the art in FTC with respect to flight control, while section 2.6 describes the use of FTC in UAVs. Finally, the last section (section 2.7) outlines the methodology and the research questions I tackle in this thesis.

2.1.3 UAVs and the Need for Fault Tolerance

There has been a gradual evolution of UAV mission roles from expendable target drones to expandable multi-mission aircraft with mission specifications extending to the more elaborate such as missile decoys, military reconnaissance, maritime surveillance, and even combat missions. UAVs are increasingly being used in front-line combat, that is, battlefield missions, to reduce human casualties. In parallel, there are many civilian applications of UAVs, in meteorological and atmospheric research, border patrol, agricultural spraying and bush fire surveillance. A characteristic feature of current UAV flight missions is the decreasing reliance on human intervention to control the flight vehicle with a significant shift towards autonomous flight whereby the flight control decisions are executed by an on-board computer according to some pre-programmed flight plan. This autonomous characteristic has been made possible with the development of advanced flight control systems and autopilots capable of operating the aircraft with minimal human input. Thus, the mission requirements have been further extended to include the capability of UAVs to fly long endurance flights in different environments.

These advanced flight control systems and autopilots are referred to as flight controllers and, under operational conditions, it is necessary for the flight controller to be robust enough to handle any uncertainties that might arise due to unexpected changes in the system parameters. Commonly used flight controllers are explicitly designed to accommodate predictable external disturbances, for example, wind turbulence, and maintain a stable flight path on the set trajectory. However, the resilience of the controller to hardware malfunction or faults is implicit, in the sense that it is a desirable feature rather than a requirement. Most controllers currently in service are not intrinsically fault-tolerant with respect to failures associated with aerodynamic control surfaces, electro-hydraulic actuators and diverse motion measurement sensors. Fault tolerance in modern flight control is achieved through the design and implementation of multiply-redundant systems, specifically through the addition of supplementary actuators and sensors, which are brought into action in the event of the failure of a member of the principal set of components. This improves general system reliability and flight safety, but incurs not only the direct cost of added hardware, but also the cost associated with the extra weight penalty and additional system complexity. This means of achieving increased fault tolerance is valid for larger conventional aircraft that can physically accommodate the multiple-redundancy hardware, however the compactness of UAVs makes multiple-redundancy of hardware impractical and costly.

An alternative solution is the design and implementation of a re-configurable flight controller that can exploit so-called analytic redundancy, as discussed in [70] and [119]. Analytical redundancy arises from the existence of inherent redundancies in the system dynamics. The focus of this thesis is model predictive control (MPC) since it has the potential (as illustrated by Kale and Chipperfield [70]) to correctly exploit these inherent redundancies.

A typical flight-control system is usually separated into outer- and inner-loops. The outer-loop guides the plane on a pre-specified path and generates a set of speed and rate demands. These demands are passed on to an inner-loop controller, which determines the requisite control surface deflections for achieving the demanded rate values. The inner-loop control structure is usually defined *a priori*, with specific functions that tell the plane how to fly. Furthermore, in the case of redundant control surfaces, some form of control allocation is performed to blend the actions of the multiple surfaces. This two-tiered process almost always assumes a fixed structure of the control, for example, the use of ailerons to control roll and the use of an elevator to control pitch. In contrast, the approach proposed in this project circumvents the need for a two-tiered, fixed structure process. In many ways, the proposed methodology is a paradigm shift in aircraft control.

In MPC, the focus is on designing a controller where the inputs into the controller design are *what to control*, instead of *how to control*. This is a subtle, but illuminating, difference, meaning inherent system characteristics such as non-linearities and cross-coupling effects can be exploited by the controller, rather than trying to minimize their influence. For example, [94], the primary function of the rudder of an aircraft is to provide yaw or sideways control. However, the rudder can also have some effect on the roll of the aircraft. Therefore, in the event of a failure of an aileron actuator, the primary control surface for roll, it is still possible to execute a limited roll manoeuvre with the rudder. In order to achieve this degree of fault-tolerance in the flight control system, a suitable re-configurable architecture is required to be purposefully designed and implemented. As part of this fault-tolerant control scheme there is a need to, initially, detect and identify the failure. Once the failure is identified, the controller must be able to compute a reconfigured/adaptive control law capable of exploiting, optimally, the available analytic redundancies, such that the effect of the failure is adequately counteracted or negated. Subsequently, upon implementation, the reconfigured controller should be able to re-establish

control, albeit with limited capacity, and execute the required manoeuvres. The mission can then either be continued with the failed component or aborted. The primary objective is to avert a catastrophic failure or the loss of the aircraft, and to ensure that it is brought back to ground safely.

The key to the design of reconfigurable control systems is exploiting the analytical redundancy of the UAV. In this context, the most promising approach to re-configurable and fault-tolerant control is MPC or variants thereof (see [70], [18], [55], [88]). Predictive control systems are designed by utilising real-time optimization techniques, where a defined objective (or multi-objective) function is optimized subject to plant operational constraints. The problem is formulated in terms of the current control inputs, states of the system and the previously computed controlled outputs. As the solution converges, the controlled outputs are expected to be the ones required to achieve the control objectives or desired trajectory. This is the context of this research.

The rest of this chapter details the current state of research in fault tolerant control systems and begins with a very brief look at flight control systems (section 2.2).

2.2 Flight Control Systems

Flight control systems (FCSs) are essentially the brains of an aircraft [42], as they control the direction of flight. Over the years there have been many variants of FCSs. Since the successful motorised flight by the Wright brothers there have been many improvements with three main types of flight control systems currently in existence [42]; mechanical, hydro-mechanical and fly-by-wire systems. The main objective of all three is the generation of the aerodynamic moments and forces required to deflect the control surfaces for flying the aircraft.

Statistics provided by Cieslak et al. [29] (2010) show that loss of control accounts for over 25% of aircraft accidents worldwide. In manned aircraft the flight control system (FCS) is operated by the pilot, however in a UAV the pilot is taken out of the loop (unless the UAV is remotely controlled). As such, in the event of a fault, the FCS must have the intelligence to handle system degradations, something that can be addressed with fault tolerant control systems, detailed in the following section.

2.3 Fault Tolerant Control Systems - A Brief History

A fault has been defined as [39]:

"an unpermitted deviation of at least one characteristic property of the system from the acceptable, usual, standard condition"

A failure on the other hand is [39]:

"a permanent interruption of a system's ability to perform a required function under specified operating conditions"

From the definitions above, a fault causes a system to behave abnormally but does not cause the system to shut down, while the latter is highly likely in the presence of a system failure. However, if left unattended, a fault may lead to a system failure, hence the importance of a fault tolerant control (FTC) system. The main characteristic of an FTC System is the ability to automatically cope with system faults before the fault turns into a serious system failure. While faults can cause instability in a system, the integration of an FTC scheme significantly increases the ability of the system to maintain overall system stability in the presence of a fault. This is of the utmost importance in safety critical systems such as aircraft, spacecraft, nuclear power plants and chemical process plants. Hence a system that has an integrated fault tolerant controller is defined as [148]:

"A closed loop system which can tolerate component malfunctions (faults), while maintaining desirable performance and stability properties, is said to be an FTC system."

Throughout the FTC literature control system faults are categorised into 3 main types [42]:

1. Actuator Faults - can mean partial or complete loss of control action.
2. Sensor Faults - malfunction of control system sensors provide erroneous measurements of the system's current status. This results in incorrect operation of the system.
3. Component Faults - all system faults other than sensor and actuator faults. These faults change the dynamics of the system as they involve changes to the physical parameters of the system due to structural damage.

In this work only actuator faults Loss of Control (LoC) and engine failures are modelled as they are the primary causes for the most serious failures in aircraft. Such faults cause changes in

aircraft parameters. It should be noted that aircraft parameters are also affected by unsteady aerodynamic and propulsive forces and moments, and are not limited to LoC conditions. Such phenomena can be seen when an aircraft is flying in the presence of environmental disturbances (such as windshear). For this reason online parameter estimation/identification is a vital part of any adaptive and reconfigurable controller, as even though the aircraft may not be suffering from LoC the aircraft will be affected. Hence any online estimation/identification techniques must be able to distinguish whether this change is due to a fault or environmental disturbance. For this work however this phenomena is not investigated as limits needed to be placed on the research.

The past three decades have seen a significant amount of research in FTC. One of the very early survey papers is the 1985 one by Eterno et. al [44]. The paper's main focus is FTC within an aerospace context and it discusses the improvement in reliability, maintainability and fault tolerance that a restructurable FCS could provide for the next generation aircraft, by taking advantage of the redundant control authority available on-board. The fault types of most concern for aircraft include those of the flight control components themselves i.e. the sensors and actuators (aerodynamic surfaces), which are usually random and are the result of battles, collisions or sabotage. Eterno et. al [44] go on to state that robustness is the goal of any control system and is defined as the ability of the aircraft to maintain performance in the presence of uncertainty. One way of achieving this is through adaptive control, the technology of continually identifying system parameters and adjusting the control parameters in accordance with the identified parameters. Continuous adaption is supported by well developed theory and numerous successful applications. If circumstances are ideal it provides graceful degradation and FCS recovery. However, most adaptive control algorithms can produce catastrophic instabilities and very high bandwidths when confronted with unmodelled dynamics and disturbance signals. In addition, the most successful applications have been on systems with long time constants and widely separated dynamics, that allow the adaptive system bandwidth to be artificially limited. The concept of the "dead zone" was introduced to address these shortcomings and involves the monitoring of the control (servo) error with the aim of determining if these errors are the result of normal command following disturbance rejection or due to plant parameter variations. Under normal operation i.e. servo errors within the dead zone, no adaption occurs; however when errors become unacceptable an adaptive scheme is applied.

Another highly influential survey paper, written by Patton in 1997 [97], classifies the FTC problem as a complex control system requiring inter-control-disciplinary information and expertise. Patton argues for new controllers that can tolerate component malfunctions whilst maintaining desirable and robust performance and stability properties. Patton [97] concurs with Eterno et al. [44] that the main requirement of an FTC system is either the maintenance of an acceptable level of performance, or graceful degradation following a malfunction. For a real time application Patton suggests the comparison of several methods on the basis of cost, robust stability, degree of predictability of the behaviour of the system and whether or not the system could degrade gracefully without loss of life/injury and/or significant economic loss. Other important factors in the decision making process include computational burden as well as the complexity of the system as a highly complex system could decrease overall system reliability. Patton comes to the conclusion that the main objective of fault tolerance should be the design of a controller able to guarantee stability and satisfactory performance, not only during healthy operations but also during component malfunctions. Such structures are referred to as control loops that possess loop integrity or reliable control. Hence FTC is a strategy for reliable and highly efficient control law design. According to Patton, research in FTC during the 1970s and 1980s has concentrated on:

- Supervision - which manages the controller reconfiguration.
- Fault detection and identification (FDI) - a very mature field that provides very powerful quantitative and/or qualitative modelling tools and artificial intelligence.
- Robust control - very popular since late 1970s. Until 1997 very little research had been carried out in regards to the effects of faults upon the control process. The few cases dealing with faults are referred to as the passive approach which will be discussed in a later section.
- Reconfigurable control - very popular amongst researchers, with methods including feedback linearisation, pseudo-inverse, adaptive control and model following, just to name a few.

Patton believed it was important to combine fault detection and identification (FDI), reconfigurable control and robust control and that the biggest challenge in FTC design was the integration of the design and implementation of a reconfigurable control scheme based on robust controller designs, and an FDI unit. He highlights the most important mathematical challenges as:

- The need for FTC to be implemented via a systematic and integrated approach to design and,
- The need to understand the structure of the system, the reliability of the different components, the types of redundancies available or those that can be generated, and the types of controller functions which might be required and are available.

Patton [97] introduces the concept of analytical redundancy (AR) as a means of using functional relationships between system variables to accommodate fault tolerant control. The underlying idea of AR is the use of the functional relationships between system signals. For many years physical redundancy was the basis for FTC, but with the use of digital computers on board aircraft AR has received a great deal of interest [36]. Here the redundancy is provided by incorporating the aircraft model into the controller; hence FTC that uses AR could be considered a model-based approach.

Zhang and Jiang's 2008 paper [148] also identifies AR as the future of FTC. The authors claim that over the previous three decades the increase in demand for safety, reliability, maintainability and survivability significantly expanded research into fault detection and diagnosis (FDD). Simultaneously research into reconfigurable fault tolerant control systems also increased.

Thus, the last three decades have seen a significant amount of research in FTC with the surveys by Eterno et. al (1985) [44], Patton (1997) [97] and most recently, Zhang and Jiang (2008) [148] giving a very good overview of the current state of the art. In summary Eterno et. al felt that the best solution to FTC was to have two separate controllers, one that handles the no fault case, and another that comes into action once a fault has occurred, while Patton in 1997 deduced that AR was the way forward in FTC. This was further recognised in 2008 by Zhang and Jiang who also identified the interaction of the FDI and FTC subsystems as an important area of research.

The literature shows that developments in FTC have been largely motivated by the aerospace industry. In particular, two commercial aircraft accidents that occurred in the 1970s highlighted the need for equipping aircraft control design with self-repairing capabilities to assist the pilot in landing the aircraft safely. In the first incident, the pilot of DELTA flight 1080 was able to reconfigure the aircraft's lateral control elements and land the aircraft safely when a fault in the elevator caused it to jam at 19^0 up. In the second, the American Airlines DC-10 Flight

191 crashed, as the pilot had only 15 seconds to react. Subsequent investigations into the latter accident found that the crash could have been avoided if there had been an FTC system on board.

As previously mentioned FTC has expanded to many industrial and academic communities due to the increase in safety and reliability demands, including aerospace [60], wind turbines [113], chemical process industry [149], automotive [47] and nuclear power [77]. Consequently, there are numerous solutions to the FTC design problem with the FTC research community classifying these designs as either *passive* or *active*. The following two subsections discuss this important distinction.

2.3.1 Passive FTC

Also known as reliable control of systems with integrity, passive FTC systems are based on fixed controllers and are designed to be resilient against known faults. They are designed using robust control techniques and are often designed for worst case scenarios. In a passive FTC system the main objective is the maintenance of the original system performance, with the biggest advantage being low computational cost as no fault detection is performed. Other advantages of a fixed controller include minimal hardware and software requirements, and with their low complexity they can be designed to be more reliable than an active system. However, since no fault detection is performed this approach is often ineffective as only a small subset of possible faults can be considered. Another disadvantage of the passive approach is that increased robustness against certain faults is only possible at the expense of decreased nominal performance.

2.3.2 Active FTC

Also known as self-repairing, self-designing or fault detection, identification (diagnosis) and accommodation schemes, active FTC systems contain an FDI component and are based on controller redesign, or selection/mixing of pre-designed controllers [42]. The FDI element monitors the health of the system, with the aim of detecting and isolating system faults. This information is then sent to the controller which reconfigures based on this new information to best represent the current state of the system. Active systems can be further broken down into two types [49]:

1. Projection based methods - based on a number of controllers with each representing a different fault. Once the FDI detects and isolates a fault a selection must be made of

the controller that best suits the current state. Like passive systems these methods can account for only a subset of faults.

2. Online redesign methods - recalculate controller parameters and are referred to as reconfigurable control. Online redesign methods have the best performance, however they are the most computationally demanding.

When comparing active and passive FTC schemes, the drawbacks of active control include false alarms, non-detection delays and complexity of control laws. On the other hand passive systems can be overly conservative and are designed to handle a certain disturbance set with the fault tolerant controller unable to handle any disturbance not within this set. Another disadvantage of passive systems is a result of utilising robust control techniques with Wu et al. [129] maintaining that a robust system often masks failures. A recent review paper written by Jiang and Xiang [68] provides an excellent qualitative and quantitative comparison between active and passive approaches. Both papers [129] and [68] suggest combining the two strategies to form a hybrid approach that can exploit the advantages of each whilst at the same time eliminating the disadvantages.

The following section briefly discusses the various mathematical control techniques that have been used in FTC. The main focus is on MPC, the method used in this research. As will be clarified later, MPC with its inherent fault tolerant capabilities can be utilised to form a hybrid FTC system.

2.3.3 Control Techniques used for FTC - Model Predictive Control

Various mathematical control techniques have been used throughout the literature for both passive and active approaches. MPC has been chosen for this research as a method for further investigation and this subsection briefly outlines MPC which is described in further detail in subsequent chapters. A table of other approaches used for FTC are listed at the end of this subsection along with their advantages and disadvantages.

Model Predictive Control - has many characteristics that make it ideal for fault tolerant flight control. MPC is an optimal control technique and is the only advanced control technique to have had a significant impact on industrial process control [90], mainly because it is the only

generic control technology that can handle system constraints.

MPC is a model-based approach that utilises an internal model to generate predictions of future plant behaviour. MPC also has excellent fault tolerant potential, however there is a reluctance to use MPC for flight control because of the use of fast time constants in comparison with traditional applications. The features of MPC that make it a promising candidate for fault tolerant flight control design are:

- Faults are easily modelled; for example a stuck actuator can be modelled by appropriate choice of constraints.
- In the event of a system fault the objective function and/or the constraints are easily modifiable as the control signal is recomputed at every time-step.
- MPC possesses particular fault tolerant properties, allowing the handling of faults up to a certain degree even in the absence of fault knowledge.

The basic idea behind MPC is to minimise a cost function subject to the system dynamics and input and output constraints. This optimisation takes place over a predefined prediction horizon. More details on MPC will be provided at the start of the appropriate chapter.

While MPC is the method of choice in this research project, for the sake of completeness it is important to consider other popular approaches to FTC. The following table lists the most commonly used approaches for FTC along with their advantages and disadvantages.

Method	Linear / Nonlinear	Active / Passive	Advantages	Disadvantages
Model predictive control MPC [70]	Both	Both	Faults are easily modelled; in the event of a system fault the objective function and/or constraints are easily modifiable; inherent fault tolerant properties exist allowing the handling of faults up to a certain degree even in the absence of fault knowledge.	Requirement of an accurate model; online optimisation is computationally demanding.
Linear quadratic regulator LQR [142]	Linear	Both	Amplitude requirements on the control inputs and the settling time of the state variables can be taken into consideration.	Constraints not easily integrated.

Method	**Linear / Nonlinear**	**Active / Passive**	**Advantages**	**Disadvantages**
Pseudo Inverse Method PIM [21]	Linear	Active	Simplicity in computation and implementation.	Stability of the impaired system is not guaranteed.
Gain scheduling GS and linear parameter varying LPV [50]	Nonlinear	Active	Utilises the advantages of linear design techniques because the basic idea is to design local linear controllers based on linearisation of the nonlinear system at several expected operation points. Computational burden is less than other nonlinear design approaches.	Linearisation scheduling is too conservative hence may not yield an effective controller. It may not be possible to cover the whole flight domain because too many operating points need to be considered.

Method	Linear / Nonlinear	Active / Passive	Advantages	Disadvantages
Linear model following LMF [91]	Linear	Passive	Design process is very straightforward and the resulting control systems have a simple structure with only the constant gains to be implemented in the feedforward and feedback path.	Important to ask the question whether perfect model following (PMF) can be achieved i.e. whether the plant can follow the reference model exactly. If Erzberger's conditions can be satisfied then PMF is achievable, however this results in severe constraints being placed on the reference model therefore making it, in many cases, impossible to achieve PMF.
Adaptive Control AC [137]	Linear	Active	Ability to improve itself given unexpected circumstances.	Robustness against noise and high frequency unmodelled dynamics is not guaranteed.

Method	**Linear / Nonlinear**	**Active / Passive**	**Advantages**	**Disadvantages**
Multiple Model MM [55]	Nonlinear	Active	Time to fault awareness can be decreased.	A number of models are used to represent a "no fault" and various fault scenarios. If the current active model does not correctly represent the current state of the system the control action provided will not be optimal.
Eigenstructure assignment EA [67]	Linear	Active	Ability to recover the eigenvalues of the pre-fault closed loop system completely.	Achievable eigenvalues and eigenvectors may not allow the system to recover.

Method	**Linear / Nonlinear**	**Active / Passive**	**Advantages**	**Disadvantages**
Dynamic Inversion DI (or Feedback linearisation FL) [117]	Nonlinear	Both	Simple nonlinear design. Easy online implementation, guaranteed asymptotic rather than exponential stability for the error dynamics and system, parameter changes are easily integrated into the control design.	An accurate mathematical nonlinear system model is required and the inverse dynamics encompassing the full flight envelope has to be evaluated; method involves a matrix inversion to calculate the control law and the existence of a matrix inverse may not be guaranteed for all time; stability of internal dynamics is not guaranteed.
Robust control [103]	Linear	Passive	Most popular method is H_∞ which is easily applied to multi variable systems.	High level of mathematical understanding is required, as is the need for a reasonably good model of the system to be controlled. Nonlinear constraints such as saturation are generally not well handled.

Method	Linear / Nonlinear	Active / Passive	Advantages	Disadvantages
Quantitative feedback theory QFT [93]	Linear	Both	Less complex than other robust control techniques. Easy to find the best controller intuitively for single input single ouput (SISO) systems. Can be applied to multiple input multiple output (MIMO) nonlinear systems.	It is an iterative procedure and the iterations can become computationally intensive when applied to MIMO.
Linear matrix inequality LMI [79]	Nonlinear	Both	LMIs are an effective tool for solving a vast number of optimisation and control problems such as linear inequalities, convex quadratic inequalities, matrix norm inequalities and various constraints from control theory such as Lyapunov and Ricatti inequalities can all be written as linear matrix inequalities (LMIs).	It may not be possible to write the problem in terms of a set of linear matrix inequalities (LMIs).

Method	Linear / Nonlinear	Active / Passive	Advantages	Disadvantages
Variable structure control VSC/sliding mode control SMC [131]	Nonlinear	Both	Sensitivity to plant parameter uncertainty is low, order of the dynamic model of the plant is reduced and as a result of the discontinuous control law convergence is within a finite time.	Implementation imperfections lead to chattering and there is too much focus on matched uncertainties.
Generalised internal model GIMC [35]	Linear	Active	Generalised internal model control (GIMC) is a compromise between conservative approaches developed for robustness and performance.	A switch is made between a high performance nominal controller and a robust controller when a fault occurs, so timely fault identification is imperative.

Method	Linear / Nonlinear	Active / Passive	Advantages	Disadvantages
Control allocation CA [34]	Both	Active	For effective reconfiguration, control allocation (CA) requires information on the health of all actuators, minimal information is required to identify a fault and allocate control authority.	Method is only appropriate for heavily actuated systems. The system is not guaranteed to remain stable and after a fault occurs the actuator dynamics and limitations are not taken into account.
Intelligent control IC [95]	Nonlinear	Both	Easily applied to nonlinear systems.	Most commonly applied method is the artificial neural network (ANN) which requires the on-line adjustment of a large number of parameters (weights) that may require very complex tuning in different flight regimes.

2.4 Fault Detection and Identification

As Patton [97] points out, one of the biggest challenges to designing fault tolerance is incorporating an appropriate fault detection and identification (FDI) unit. FDI is a very mature area of study and provides many powerful quantitative and qualitative modelling tools yielding artificial intelligence regarding any given fault. The FDI module is an integral part of the active FTC system and can mean the difference between complete system failure and system recovery. FDI is the most difficult aspect of FTC [26] and due to the excessive costs involved and increase in system weight, hardware redundancy and full self diagnosis equipment are not always viable solutions, particularly for UAVs. As a consequence other means for FDI are necessary, such as using the available data and the mathematical model of the plant; this is referred to as model-based FDI, and as analytical redundancy. This approach to FDI is based on the belief that when a fault occurs the physical parameters change and as a result the dynamical model of the plant also changes. The residual generation approach, using observers such as the Kalman filter, the Unscented Kalman Filter and Multiple Model Filters, is the most researched area in FDI [8] and is investigated as part of this research. Observer based schemes are highly dependent on the models upon which the scheme is designed. False alarms or missed faults can occur due to plant model mismatches, hence robustness issues in FDI are critical.

Patton [97] identified the role of an FDI system as the ability to gather information about the changes occurring in the system parameters (or in the system operating point) due to faults. There have been many kinds of FDI approaches developed [97]; quantitative, qualitative and knowledge based. Quantitative FDI can be further classified as state estimation, parameter estimation and parity equation approaches. However, FDI research has more commonly been conducted in isolation to controller design and there is a huge gap in integrated FTC and FDI design. The reliability of the FDI system is required to be higher than the system being monitored as the better the model used to represent the dynamic behaviour of the system, the higher the chance of improving the reliability and performance of detection and isolation of faults thereby reducing the number of false alarms.

Fault detection is clearly an essential component of any active FTC system and hence will be thoroughly examined in chapter 4.

2.5 Fault Tolerant Flight Control - The State of the Art

The many methods of control techniques given in subsection 2.3.3 have been used in fault tolerant flight control design by many authors. In this section, I detail the use of these methods providing examples to illustrate their variety.

The past decade has seen a great deal of work in the area of fault tolerant flight control. For instance the European Flight Mechanics Action Group FM-AG(16) on FTC was in operation from 2004 to 2008. A collaboration of thirteen European partners from industry, universities and research establishments, FM-AG(16) was supported by the Group of Aeronautical Research and Technology in Europe (GARTEUR) program [42]. The focus of this latter group was commercial transport aircraft because statistics clearly indicated that many airliner accidents could be attributed to Loss of Control In-Flight (LOC-1), caused by a piloting mistake (e.g. due to spatial disorientation), technical malfunctions or unusual upsets due to external disturbances. Hence the aim of the GARTEUR Flight Mechanics Action Group FM-AG(16) with regards to fault tolerant flight control was the development of new fault tolerant control strategies within the European aerospace research community in practical and real-time operational applications.

SImulation MOtion and NAvigation (SIMONA), a 6-DoF flight simulator provided by Delft University (Netherlands) was used to assess the real time applications of the FTC technologies developed by the FM-AG(16) group. The research undertaken by the FM-AG(16) group, as given in [5] and [6] includes an FTC scheme based on sliding mode control (SMC) and control allocation (CA). CA has emerged as one of the leading techniques for dealing with systems with redundancy such as large transport aircraft. A benefit of CA is that reconfiguration of the controller is not required as CA schemes "automatically redistribute" the control signal. Sliding mode control (SMC) is a non-linear control methodology and comes under the category of robust control hence it has robustness properties against certain types of disturbances and uncertainties that make it suitable for FTC. However, since sliding mode control (SMC) cannot deal directly with actuator failures, CA offers a solution by providing access to the redundant actuators. Simulator results obtained by the FM-AG(16) group were good even in wind and gust conditions. Alwi and Edwards in an earlier paper [7] presented the SMC/CA scheme applied to the lateral and longitudinal axes of the non-linear B747 aircraft simulation model where they showed that the sliding mode control allocation scheme could handle fault and total actuator failures directly without reconfiguring the controller. Other formulations of FTC that have come

out of the FM-AG(16) group include an adaptive nonlinear dynamic inversion configuration for manual and autopilot control [84], where the authors explain that the weakness of classical nonlinear dynamic inversion (DI), sensitivity to modelling errors, can be avoided by the use of a real time identified physical model of the damaged aircraft.

Throughout the literature it was found that FTC schemes are most commonly a combination of the methods outlined in section 2.3.3. The aim of the Intelligent Flight Control System (IFCS) F-15 program at National Aeronautics and Space Administration (NASA) is the development and evaluation of flight control schemes that assist the pilot in handling faults during the occurrence of a primary control surface failure. The FTC scheme developed under this program ([101], [99]) is based on nonlinear dynamic inversion and is augmented with a neural network (NN) to compensate inversion errors and changes in aircraft dynamics due to damage or failure of a primary control surface. Three different NNs were investigated, the Extended Minimal Resource Allocating Networks (EMRAN), Single Hidden Layer (SHL) and the Sigma Pi. The simulations were conducted using the NASA F-15 aircraft and different fault scenarios of the stabilator and canard were investigated. Results showed that EMRAN outperformed the other two schemes in terms of angular rate tracking errors while requiring lower computational effort.

Other examples of combined methods for FTC include work conducted by Shin and Gregory [112] where the FTC method is based on robust gain scheduling (GS) control concepts using a linear parameter varying (LPV) control synthesis method to design fault tolerant controllers for a civil transport aircraft. Both passive and active controllers were designed and implemented for the longitudinal motion of the aircraft and the effect of time delays was investigated. It was found that in the active FTC case controllability was not guaranteed for fault detection delay times greater than ten seconds. The passive controller on the other hand used the elevator and stabilisor for control even in healthy situations where only the elevator should have been used.

Yang and Lum [134] tested a solution for FTC on simulation models of the F-16 aircraft with stuck actuator faults, with the FTC based on H_∞ and peak-to-peak gain performance indices in a multiobjective optimisation setting where the algorithms were based on linear matrix inequalities (LMIs). Zhang and Jiang [147] on the other hand, tested an active FTC design on a model of the F-8 aircraft, finding that in the presence of a fault the degradation in dynamic performance was accounted for through a degraded reference model. Zhang and Jiang [147]

used an eigenstructure assignment (EA) algorithm to automatically design the controller once a fault was detected. This was done in a model following framework so that the dynamics of the closed-loop system follows that of the degraded reference model.

Yu and Jiang [143] developed an active FTC method where the impairments were modelled as a polytopic LPV system. Here, simulations look at inner elevon impairments and the FTC was based on LPV techniques through the optimisation of linear matrix inequalitys (LMIs). Ye et. al [139] used a linearised model of the F-18 (longitudinal motion only) where the FTC was based on H_∞ in an LMI framework similar to Yang and Lum's work [134]. Ye et. al had earlier produced a two part FTC system [140] where the first part consisted of a nominal controller based on iterative LMI. The second part of the system was a re-allocation scheme that redesigned an optimal control law without reconfiguring the baseline controller and was based on the pseudo inverse method (PIM). Simulations were performed on ADMIRE (the Aerodata Model in Research Environment), a non-linear, six degree of freedom simulation model of a rigid small fighter aircraft with a delta-canard configuration developed by the Aeronautical Research Institute of Sweden. A more recent paper by Gao and Wang [51] focuses on FTC for an air breathing hypersonic vehicle (AHV) subject to actuator faults and limited measurements of the states where the scheme is based on feedback linearisation (FL) and SMC.

2.5.1 Sliding Mode Control Approach

Sliding mode control (SMC) was found to be one of the most popular methods for FTC in the literature. Alwi and Edwards [8] developed a sliding mode approach for FTC of a civil aircraft (model of the Boeing 747) where both actuator and sensor faults were considered. In [8], a state feedback sliding mode controller is designed for actuator faults where the gain of the nonlinear unit vector term is allowed to adaptively increase once a fault occurs. The adaption mechanism is activated by unexpected deviations of the switching variables from their nominal condition. A robust method for fault reconstruction using sliding mode observers is used for sensor faults. The proposed method does not require controller reconfiguration because the corrupted measured signals are corrected via a sensor fault reconstruction signal before they are used by the controller. In the nonlinear part of the control law, an adaptive gain is used which reacts to the occurrence of a fault and attempts to keep the switching function as close as possible to zero, thus trying to maintain nominal tracking performance. A switch is made to a back up control

surface if total failure is detected. However, the linear component of the control law remains unaltered. The novelty of this scheme is the design of the hyperplane, which minimises the effect of unmatched uncertainty on the sliding motion arising from actuator failures and the simple adaptive scheme for the nonlinear unit vector scaling gain. Hence the FTC controller is based around a state feedback sliding mode scheme and the gain associated with the nonlinear term is allowed to adaptively increase when the onset of a fault is detected.

Another application of SMC can be found in [49] where a passive FTC scheme is developed for the longitudinal motion of an F-18A aircraft. Pre-specified faults are handled via a variable structure controller with a sliding surface and a Lyapunov function. The authors claim that the main features of their proposed method are simplicity and robustness against uncertainties and parameter variations due to some pre-specified faults. Peng et. al [98] have produced an adaptive and conditional integral SMC. As the authors points out, SMC suffers from chattering which can excite unmodelled high-frequency dynamics, degradation of system performance and result in instability. The authors develop an SMC formulation that reduces chattering and guarantees zero-steady state error achieving this through a conditional integral term to attain steady state error and an adaptive technique based on Lyapunov stability theory to compensate for the effects of the disturbance generated by actuator faults.

Yang et. al. [135] also produce a passive FTC scheme based on adaptive SMC again using the Lyapunov stability theorem, to ensure closed loop stability. Comparing adaptive SMC to classical SMC, they found, through simulations of aircraft actuator faults, that adaptive SMC performs better than the conventional SMC fault tolerant controller. The very same group also produced [136] a model following based SMC controller, where proportional integral type sliding surface was designed via pole placement to ensure the sliding mode had good dynamic characteristics and to eliminate steady state error.

2.5.2 Multiple Model Techniques

Multiple Model techniques are another very popular method for FTC, however these are primarily used as a means of predicting faults. Boskovic [17] presents a decentralised FDI and adaptive reconfigurable control scheme to achieve the desired flight performance in the presence of multiple control effector failures that occur at different times. This decentralised scheme consists of

multiple adaptive FDI observers and controllers and a suitably chosen decision making mechanism. The proposed method [17] was implemented on a model of the F/A-18A aircraft and the decentralised system consisted of 3 subsystems: (1) FDI subsystem, (2) parameter estimation subsystem and (3) decision making subsystem. In the event of an effector failure the decision making subsystem uses estimates from the parameter estimation subsystem online to reset all FDI observers and controllers to the nominal regime which is established immediately following the failure. In a later paper of Boskovic [24], only the FDI is decentralised, i.e. an observer is run at each of the actuators; and parameter estimations are adjusted using only local information. A key aspect of this approach is that the observers do not use global state information, but rather local signals. The main advantage of this method is that it can handle multiple simultaneous faults and the controller uses parameter estimates from each of the observers.

Guo et. al [61] introduce an active FTC scheme based on a combination of a direct adaptive control algorithm with multiple model (MM) switching. The FTC in [61] is based on radial basis function networks (RBFN) type neural networks to approximate model uncertainty and adaptive parameters, while MMs are used to describe all fault scenarios.

Intelligent control techniques have been widely used for FTC. Liu et. al [83] developed a passive FTC scheme based on an NN model following adaptive inversion control that uses an adaptive back propagation NN. Li et. al [78] describe an FTC technique for autolanding based on a radial basis function network (RBFN) and traditional controllers. Due to the powerful ability of NNs to approximate nonlinear functions they are able to adapt to changes in system dynamics quickly while still providing good performance hence in [78] FTC is based on an NN aided H_∞ controller. Yan et. al. [133] proposed an FTC method using a nominal controller plus an adaptive controller based on NN, in particular the minimal radial basis function neural network (MRANN), and were one of the first people to use MRANN on flight control. Simulations were based on the longitudinal motion of the F8 aircraft. Other applications of NN based FTC can be found in [92], [109], [146]. The fuzzy logic controller is used by Sami and Patton [110] where the goal was the design of a controller that could handle simultaneously occurring actuator and sensor failures. The methodology in [110] is based on a TS fuzzy controller and TS fuzzy observer. Torabi et. al [118] compare a fuzzy controller with MPC as fault tolerant controllers for the pitch rate command tracking of light aircraft. Results show that the effect of disturbances is better handled by MPC.

H_∞ has proven to be a very popular robust control method for FTC. Cieslak et. al [29] use an H_∞ base controller for the landing approach of the B747-100/200. Piloted flight simulations analyse a faulty trimmable horizontal stabilisor and show that fault tolerance can be achieved under the condition that there exists sufficient remaining control authority. Wu and Chen [129] also incorporate H_∞ for FTC where the aircraft considered has redundancy in control authority provided by both elevons and canards. Ye et al. [138] look at mixed H_2/H_∞ robust FTC, with the FTC set up in an LMI framework where a multi-objective optimisation problem is solved. Simulations are performed using the ADMIRE simulator. The H_2 controller is adapted to handle the transient performance while the H_∞ guarantees robust stability in the presence of uncertainties and disturbances.

Adaptive Control techniques have been used by Boskovic et. al [19] where the approach is based on generating high frequency signals for actuators with suspected failures with the aim of minimising the effect of these signals on the system state using the remaining healthy control surfaces. This method was tested on simulations of the F/A-18. The preliminary results were quite positive showing that the proposed technique was robust to false alarms, false failure information and missed detections and it assured convergence of the failure parameter estimates to their true values. Boskovic et. al also produce a method of adaptive control [16] for FTC where a suitable tracking error feedback (TEF) term is designed such that the plant is stabilised over the entire uncertainty set. If such a term exists then the adaptive control part depends only on the reference model state and reference input. This results in a stable linear time-varying system rather than a nonlinear time-varying system which commonly arises in the context of standard adaptive control (AC) and allows for simpler analysis of the system. Other applications of AC can be found in Gayaka and Yao [53] and Idan et. al [65] where, in both cases, model reference adaptive control is used as the basis for FTC.

Other methods that have surfaced in the area of FTC include using off-line multi-objective optimisation [14] to design an optimal gain tolerant to a control surface failure. This method can only be applied to a known set of failures and in the paper [14], is applied to a model of the F/A-18A aircraft. Ciabotaru and Staroswiecki [30] use a linear quadratic regulator (LQR) design for FTC of the short-period mode of the longitudinal model of the Boeing 747. When actuator faults occur, both fault accommodation and system reconfiguration procedures need the

controller to be redesigned which means solving algebraic Riccati equations (AREs) associated with the post fault model. To avoid the risks of system instability and/or control inadmissibility which results from fault detection delays, the authors [30] utilise a progressive accommodation strategy based on the Newton-Raphson algorithm for solving AREs.

An analytical based FTC approach can be found in a very early paper written by Gross et. al in 1986 [59], which points out that valuable information can be obtained about the status of an actuator by examining the actual versus commanded position. Gross et. al suggest the use of hinge moment residuals as an approach that could yield valuable information for damage detection, isolation and estimation of effectiveness. Keating et. al [71] present a quantitative feedback theory (QFT) based FTC system. QFT uses a 2DoF arrangement that utilises unity feedback, a cascade compensator and a prefilter to reduce variations of the plant output from plant parameter variations and disturbances, rendering it necessary to make trade-offs between compensator complexity and performance.

One area of FTC that has not been thoroughly researched is the hybrid method, a combination of the active and passive methods [148]. Fekih [48] developed a FTC method based on integrated (active-passive) design combining SMC with adaptive control with simulations carried out on a model of the F-16. Here the sliding mode method is the nominal controller; if its performance degrades such that the state evolves outside a given boundary then the adaptive controller is added to the nominal controller to offset the actuator faults. Yu et. al [145] also propose a scheme to combine active and passive approaches with the basic controller based on LMI. When a fault occurs initial fault detection is performed by the autonomous robust reliable control system and control is set to a robust control law selected from a set of reliable control laws proposed off-line. In the background the FDI system is simultaneously reconfirming the fault, and after it has been isolated the reconfigurable control law is developed. Simulations were conducted on a helicopter rather than a fixed wing aircraft and the results showed that the system could achieve optimal performance under the nominal condition and in the event of a failure the system responded instantly, guaranteeing stability and a certain level of performance. Finally Yu and Jiang [144] design both active and passive components of their hybrid FTC scheme via LMI, describing the means by which a hybrid FTC system is able to first slow down the rate of fault induced system deterioration with minimal fault information such that the FDI subsystem has a chance to correctly isolate the fault. Once detected the reconfigurable controller takes

over from the passive robust controller. Through simulations, using both linear and nonlinear case studies, Yu and Jiang [144] found that passive systems use more energy than active systems. They also found that with correct FDD information the active system performance is superior compared to the passive system. The passive system was found to be very conservative.

This section covered the state of the art in fault tolerant flight control. It discussed the techniques presented in section 2.3.3 as applied to flight control for the purposes of fault tolerance. The next section takes a more in depth look at the techniques that have been applied specifically to UAVs, the main application area of this research.

2.6 UAV FTC Methods

As is evident from the literature review outlined so far, especially in the previous section, fault tolerant flight control has been mainly utilised within the context of large manned aircraft. Much of the literature on UAVs describes the application of FTC to rotorcraft rather than fixed wing aircraft. This section provides a brief overview of the work being conducted in UAV fault tolerant control.

Bateman et. al. [11] believe that gaining airworthiness approval for UAVs in civil airspace requires an increase in reliability and propose an active FTC system to deal with control surface failures for a UAV. The fault tolerant control scheme in [11] consists of an FDI method based on a signal processing approach and a bank of linear quadratic controllers to handle all faults. Fault detection is carried out by first detecting a fault, followed by an isolation process whereby each control surface is excited with a specific signal that constitutes its signature. Hence isolation of the fault involves identifying the presence or the absence of this signature. The authors explain that in the presence of an actuator failure the equilibrium of forces and moments is broken and significant couplings appear between the longitudinal and lateral axis of the aircraft. To find a new equilibrium the fault free mode control surface deflection constraints are relaxed. FTC systems exploit the redundancies offered by the control surface and control each one separately, via a linear quadratic controller. A drawback of the proposed method in [11] is that it does not allow faults to be detected in the rudder due to the lack of redundancy in this control surface.

In 2008 Bateman and the same co-authors [12] presented a paper investigating an FTC strategy for the nonlinear model of a UAV equipped with numerous redundant controls. Here the

authors look at asymmetric actuator failures using a sequential quadratic programming (SQP) algorithm that takes into account nonlinearities, aerodynamic and gyroscopic couplings, state and control limitations as a means for FTC. The algorithm calculates new trim conditions such that the new operating point of the faulty linearised model remains near the fault free model. The authors [12] examine asymmetric failures for which couplings appear between axes, possibly changing the equilibrium of forces and moments. The time required to process the failure may move the state vector far away from its operating point, so a nonlinear model of the aircraft that takes into account aerodynamical effects of each control must be considered. Under this scheme an FTC system can be implemented if and only if a new operating point can be found. To compute this new operating point the authors assume that the faulty controls and their positions are known, i.e. FDI is assumed. Re-allocation of the healthy actuators is expressed as an optimisation problem with equality and inequality constraints. The optimisation returns the new operating point in faulty mode at which point it is possible to calculate a linearised model of the UAV. Upon computing the new operating point a linear state feedback is calculated which aims to steer the current state vector towards equilibrium. Hence in theory one controller has to be designed for each fault situation; to achieve this the authors [12] use EA as the basis of the controller design.

Other applications of FTC for UAV operations can be found in very early papers on the subject by Copeland and Rattan (1994) [32], Chen et. al (1998) [28] and Wu et. al (1999) [130]. Copeland and Rattan [32] propose a fuzzy logic supervisor algorithm for a reconfigurable flight control law. The authors claim that in the event of a fault the set of fuzzy logic rules obtained ensure even distribution of control authority to the remaining healthy effectors. The FTC system described by Chen et. al [28] (1998), uses LMI to address wing impairment faults on UAVs. The authors present a multi-objective approach for establishing a matrix inequality formulation. The approach is designed to eliminate the rank constraints in the LMI that occur in the presence of a fault, which if left unattended may lead to a non-convex problem. Wu et. al [130] use QFT as the method of choice for FTC on a remote pilotless aircraft. The approach described therein [130] was applied to the longitudinal motion of the aircraft and results showed an increase in stability and tracking performance compared to the existing FCS.

In more recent work by Beainy et. al [13] FDI is based on NN and a reconfigurable controller based on SMC is designed to compensate for the degradation of the actuation on the occurrence

of a fault. Krueger et. al [76], on the other hand, present an FTC based on an expanded nonlinear model inversion flight control strategy using sliding mode online learning for NN. Another NN based FTC system was presented in [102] where the proposed system had three main components: (1) an FDI, (2) a controller suite comprising of a nominal controller and an NN based adaptive fault tolerant controller, and (3) a reconfiguration supervisor that makes decisions regarding controller reconfiguration. In [122] fault tolerant control is done via a standalone compensator which is added to the original system. NNs are used to design the compensator and it comes into effect after a fault occurs; it also consists of a state observer. Results show that in the event of a disturbance due to a fault the new approach improves the closed loop response.

Fan et. al [46] present a hybrid active/passive approach to a flying wing UAV using an LMI framework, discussing the importance of intelligently integrating the FDD system into the FTC system as otherwise any delays in fault detection will result in system instability. The authors claim that the unique feature of the proposed hybrid method is its ability to tackle the actuator saturation problem, allowing full utilisation of the actuator power. If and when the effectiveness of the actuator is at the limit for which tracking requirements cannot be met, then recovering the maximal tracking performance is possible if the remaining actuator power can be fully utilised by the system. The authors derive a set-invariance condition to guarantee the stability of the post fault system, designing the reliable controller such that the resultant invariant set includes the output from the nominal controller. However, stability is only guaranteed under the assumption of zero detection delay.

Ahn et. al [2] look at an adaptive and sliding mode scheme for FTC. The merits of adaptive and SMC are that (1) the magnitude of sliding mode controller gain can be reduced and (2) an FDI process is not required. Research in [2] is focused on UAVs; however simulations are on a generic 6DoF nonlinear aircraft model. Using the time-scale separation principle both the fast inner-loop states and slow outer-loop states are simultaneously controlled. An online adaptive parameter estimation scheme is considered in the pure sliding mode controller to overcome the large control authority requirements and chattering problems of SMC.

2.6.1 Damage Tolerance Control

A series of flight tests were performed under the DARPA-sponsored Damage Tolerance Program [69] in which four key technologies are discussed and illustrated with actual flight data. The Rockwell Collins damage tolerant control (DTC) technology is designed to mitigate common UAV failures such as primary control surface damage, airframe damage and complete engine failure. A combination of all types of attitude control, MRAC, automatic supervisory adaptive control (ASAC) and emergency mission system (EMMS) is shown to provide UAVs with an unprecedented robustness to otherwise catastrophic failures. From April 2007 to June 2010 flight data was collected on flights using a subscale F-18 UAV. The MRAC method was able to recover the baseline controller after losing actuation, while ASAC returned an aircraft with catastrophic wing damage to trimmed and controllable flight within seconds allowing the mission to be completed with a successful autonomous landing. The EMMS allowed the aircraft that had suffered complete engine failure to glide back onto a feasible landing trajectory.

2.6.2 Use of MPC in the Process Industry

As mentioned before, the control methodology that I study in this thesis has been extensively used in the process industry. Boskovic and Mehra [23] present an MPC based fault tolerant control scheme for the process industry where MPC controllers are found to be sometimes better suited to a given problem over proportional integral derivative (PID) controllers, since the latter cannot take into account process characteristics such as nonlinearities, time variations, loop interactions and constraints all of which are possible with MPC. The same authors also apply MPC to fault tolerant flight control in [22] where simulations are based on Boeing's Tailless Advanced Fighter aircraft, with FDI based on multiple model switching and tuning (MMST). Camacho, Alamo and de la Pena [26] detail the extension of the receding control strategy of MPC to the case of system identification by parameter bounding. Furthermore the authors show that MPC can be used to determine if a model is consistent with the data obtained in a receding horizon manner which implicitly enables fault detection. The paper by Camacho et. al [26] shows the incorporation of concepts arising from fault detection and fault tolerant design methods into an MPC framework and lists the advantages gained by utilising MPC as a fault tolerant control mechanism.

The versatility of MPC is illustrated in [26], where the authors give a formulation of MPC taking

into account faults and uncertainties that allow the design of intrinsically safe controllers, making this formulation of MPC a type of passive FTC. Camacho et. al [26] point out that because MPC can be used for set membership estimation it can also be used for FDI. Once a fault has been detected MPC is capable of coping with the new situation by using the fault mode. Thus MPC when used as a set membership estimator could be said to belong to the class of active FTC.

Applications of MPC for fault tolerant flight control date back to 1998 when Gopinathan et. al [56] used MPC for fault tolerant control, with MMST for fault detection, applying their formulation to simulations of the F/A-18A aircraft. Henson [64] proposes fault tolerant control methodology for quad-rotor flight control, explaining that most active FTC need a post-fault model updated by FDI followed by a reconfiguration process to handle the faults. However, since post fault time is crucial Henson suggests combining FDI with the reconfiguration process. MPC offers the most promise in this regard, as it calculates the control signal at every sampling time. Henson clarifies that the biggest drawback of MPC is the need for an explicit model. Henson's paper [64] offers a data-driven MPC architecture that performs model identification and control signal calculation simultaneously using available post-fault input/output data. The proposed architecture requires no reconfiguration, switching or online retuning of parameters and the identification cost is added to the cost function.

Having looked at previous research into the use of FTC in UAVs, I now outline the methodology I used in my research and the research questions that were formulated.

2.7 Proposed Methodology and Research Questions

Guo, Zhang and Jiang [61] identify many difficulties with a number of FTC systems. The main difficulty with Dynamic Inversion, for example, is that an accurate mathematical nonlinear system model is required and the inverse dynamics encompassing the full flight envelope have to be evaluated. Neural Networks, on the other hand, require highly complex tuning for different flight regimes due to the fact that they require on-line adjustment of a large number of parameters (weights). Finally, standard adaptive control algorithms are found to require strong and often unrealistic assumptions, such as, known relative degree, minimum phase etc. In my research I will investigate MPC further, as a viable FTC solution for UAV flight control.

To establish the context of my methodology, I start with some details of existing research into the flexibility of MPC. Maciejowksi, a prominent figure in the area of MPC for over two decades, describes [86] the concept of daisy chaining, which occurs in systems with redundant actuators and refers to the arrangement in which one actuator (manipulated variable) is used in normal operations but if this actuator were to become saturated or to fail, another or several others are brought into operation. In other words daisy chaining refers to the transfer of control action from faulty actuators to healthy ones. A very important point highlighted by Maciejowski [86] is that there is implicit daisy chaining capability inherent in MPC and hence it can exploit any available redundancy even in situations unforeseen by the designer. Daisy chaining would greatly enhance the robustness of constrained predictive control schemes in the event of actuator saturation as well as increasing the tolerance to certain kinds of failures.

Following this Maciejowski [88] presented statistics surrounding the Vietnam war where the US Air Force lost about 10,000 aircraft with 20% from damage to the control system. This indicates that the engine and airframe were basically undamaged, and the aircraft may have been recoverable if the pilots had been able to work out how to do so. In another paper that same year, Maciejowski [89] had proposed that MPC offered a promising basis for FTC due largely to the fact that MPC relies on an explicit internal model. This he believed, was plausible because failures could be dealt with by updating the internal model and then allowing the on-line optimiser to work out *"how to control"* the system in its new condition. This however relies on several assumptions not the least of which are:

1. The location of the fault can be determined and the effects can be modelled.
2. The model can be updated automatically.
3. The control objectives can be left unaltered after the failure.

Maciejowski [88] emphasised the importance of having a reliable FDI, and showed the importance of MPC to fault tolerant flight control in [87] when he and Jones demonstrated that the fatal crash of EL AL flight 1862 could have been avoided by using MPC based fault tolerant control. In [87], Maciejowski showed via simulation of a detailed nonlinear model that it would have been possible to reconfigure the controller so that the aircraft could have been flown down to ground level successfully without entering the condition in which it was lost. The FTC method used by Maciejowski [87] constituted a reference-model based approach in which an MPC controller attempts to restore the original functionality of the pilot's controls. Simulations were based on the

assumption that an FDI system delivers information about the actuator damage and about the change in the aerodynamic coefficients in the failed condition and Maciejowski claimed, "MPC can provide effective solutions for fault tolerant control" [87]. Maciejowski maintains that MPC is a good framework for FTC because many types of aircraft failures can be handled online in an adaptive fashion via modifications to the internal model. It is stated in [87] that the achievable performance of an aircraft will often be reduced after a failure but this too can be tackled via MPC through modifications to the objective function or the use of a multi-objective formulation. Maciejowski stresses that FTC should be used in conjunction with the pilot rather than replacing the human operator, whereas the goal of my research is to take the human operator out of the loop for true autonomy, at least for autonomous UAVs.

Unfortunately, despite the various FTC methods being researched they have not been readily adopted by the aerospace industry [80]. The level of maturity of this research is not high enough for safety critical flight especially for all flight regimes. Other contributing factors are the complexity of the designs and the very high likelihood of false alarms due to large modelling uncertainties and/or disturbances. The majority of the methods are also based on linear control techniques. The ever increasing demands placed on UAVs, particularly by the military, will require an expansion of current flight envelopes, therefore it will be necessary for control systems to evolve from simple linear based formulations to non-linear adaptive procedures.

Following this extensive literature review the following research questions were developed and form the foundation for this research:

1. Is MPC applicable to UAV fault tolerant flight control?
2. Is nonlinear MPC a viable solution for fault tolerant flight control?
3. Can MPC reach a solution within an acceptable time frame for aircraft control?
4. In the event of a fault how sensitive is MPC to controller tuning parameters such as the prediction horizon and cost weightings?
5. Which method for FDI will integrate easily with an MPC controller configuration?
6. How sensitive is MPC to delays in fault detection?
7. How robust is MPC to actuator faults?

8. Can MPC truly be considered a hybrid FTC scheme?

The results of my research into these questions will be presented in the remaining chapters of this thesis, beginning with a detailed look (chapter 3) at model predictive control (MPC), the research leading to it and the current state of this research.

Chapter 3

Design of a Nonlinear Model Predictive Controller

3.1 Introduction

3.1.1 Motivation

Model predictive control falls under the category of advanced control techniques and is classified as an optimal control method. Chapter 3 is devoted to exploring numerical applications of model predictive control (MPC), and in particular nonlinear model predictive control (NMPC) and both theoretical and numerical methods are utilised.

The potential of MPC as a basis for a fault tolerant control (FTC) system for aircraft was recognised by Maciejowski [87]. MPC has inherent fault tolerant capability for certain fault conditions and, because of the internal model used for prediction, MPC can be implemented as a reconfigurable controller. This makes it ideal for both passive and active implementations.

3.1.2 Outline

Before developing an active FTC system using MPC a thorough understanding of the theoretical and practical aspects of MPC is necessary. For this reason section 3.2 has been dedicated to exploring the theory behind model predictive control. As MPC is the major focus of this thesis, section 3.2 provides a brief history and discusses the various characteristics of MPC. A detailed description of the internal workings of MPC are provided followed by a discussion on NMPC.

Furthermore the implementation aspects of MPC are explored in section 3.3. This section focuses on the methods commonly used for the conversion of a continuous time system to a discrete time system. The objective of this chapter is to summarise the methods used in implementing optimal control techniques.

Section 3.4 presents an implementation of a number of the optimal control techniques discussed in section 3.3. The methods are applied to the well known Brachistochrone problem, posed by Bernoulli back in 1696 [40]. The Brachistochrone problem is a nonlinear problem for which an analytical solution exists, providing a means of checking the validity of numerical implementations.

The findings from the Brachistochrone problem are then used to implement NMPC using a simple 2D robot model as an illustrative example in section 3.5. Both the open loop and closed loop problems are addressed. The purpose of section 3.5 is the exploration of the practical implementation issues associated with NMPC. Comparisons are also made to linear MPC.

Finally, in section3.6, the findings are summarised and the conclusion given.

3.2 Model Predictive Control

Model Predictive Control, also known as Receding Horizon Control, is an advanced control technology developed by practitioners within the industrial process industry, that has had considerable impact on industrial process control. The main reason for this is that MPC is the only generic control technology capable of handling equipment and safety constraints [90], allowing systems to operate at or near constraints, yielding a more efficient and profitable operation. It took almost 20 years, for the academic community to pay serious attention to the theoretical aspects of MPC.

Maciejowski and Jones were one of the first to recognise the value of MPC presenting (in 2003) [87] a numerical example showing that the 1992 crash of El Al Flight 1862 could have been prevented by using MPC-based fault tolerant control. While the authors [87] do not claim to have solved the problem of fault tolerant flight control, they do claim that MPC has great potential for fault tolerant design. This, combined with the fact that MPC is a model based approach capable of handling nonlinear models, makes it an ideal candidate for further investigation as an FTC controller.

This section details the underlying ideas behind MPC and its workings. A linear process model has most commonly been used in MPC, however interest in NMPC is increasing. Hence aspects of both linear (section 3.2.1) and nonlinear MPC (section 3.2.2) will be discussed followed by practical applications of both methods to aid in the understanding of this advanced control technique.

3.2.1 Linear Model Predictive Control

MPC is a model based optimal control methodology. Unlike many control systems where the model of the plant is used only for design and analysis purposes, in MPC the model is an integral part of the control algorithm. The model is used to predict future behaviour of the plant in order to calculate the optimal control trajectory.

3.2.1.1 How Does MPC Work?

Time domain, input/output, step or impulse response models were all used in early MPC work. Today linear models are more commonly represented in state space form.

Linear MPC represented in state space form has the following advantages:

- Multivariable systems are easily handled.
- Closed loop properties are easily analysed.
- Online computation is possible.
- Linear systems theory such as linear quadratic regulator (LQR) and Kalman filtering can be used.

Figure 3.1 illustrates the workings of MPC. The MPC controller has an internal model used to predict the behaviour of the plant over a future prediction Horizon, H_p. The idea is to select the best input that will produce the best predicted behaviour. A number of coincidence points are placed over the horizon, distance k time steps apart and the aim is to bring the predicted output as close as possible to the reference trajectory. This is achieved by optimising a cost function, commonly a quadratic cost which is solved via quadratic programming in the case of linear MPC. Only the first input of the calculated trajectory is applied to the plant and the prediction window slides along by the sampling time, t_s, where t_s is much smaller than H_p.

Once the window slides to the next time step and the calculated input is applied to the plant, the new plant states are fed back to the controller and the whole cycle begins again. The length of the prediction window remains fixed but slides forward by one sampling interval at each step; a process referred to as the receding horizon strategy. To reduce the computational burden a control horizon, H_u, can be defined which is smaller than H_p. The control inputs are calculated only along the control horizon, beyond which point the value of u remains constant. Performance stability increases as the length of H_u approaches the length of H_p.

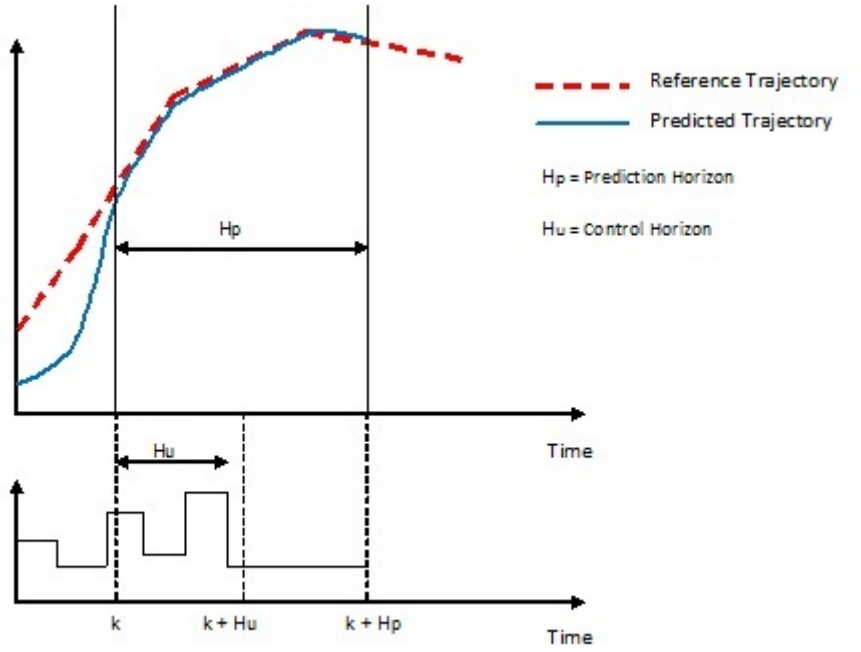

Figure 3.1: Linear MPC

3.2.1.2 MPC Techniques

There are many variants of MPC, however the overall process is the same. The models used in MPC are often black-box, linear input-output models developed via plant tests or through system identification methods applied to plant data. Some variants of the basic MPC technique [1] are listed below along with their main features:

- Model algorithmic control (MAC) - initially called model predictive heuristic control (MPHC),
  - Uses an impulse response model.

  - Is valid only for open-loop stable processes.
  - Variance of the error between the output and a reference trajectory, computed as a first order system, is minimised.

- Dynamic matrix control (DMC),
  - Similar to model algorithmic control (MAC), however a step response model is utilised.
  - Techniques employed by Shell Oil as early as 1973 come into this category.
  - Input and output constraint handling using quadratic programming is incorporated leading to quadratic dynamic matrix control (QDMC).
  - Method can also be derived for a general discrete state-space model.

- Extended prediction self adaptive control (EPSAC),
  - Model is based on either discrete with z-transform or continuous with s-transform.
  - Control law structure is simple and is calculated analytically.
  - Disturbances are included in the process model.

- Generalised predictive control (GPC),
  - Uses a quadratic performance function with weighting of control effort and an auto-regression moving average with exogenous variable model.
  - Able to provide an analytic solution for optimal control in the absence of constraints.

The most popular methods in current use are dynamic matrix control (DMC) and generalised predictive control (GPC) [1]. GPC has been modified over the years to guarantee stability through end-point equality constraints by stabilising the process prior to the objective function optimisation. Most of the work in GPC has been carried out in discrete time, however continuous time models have also been developed.

3.2.1.3 Advantages and Disadvantages of MPC

MPC is very popular in industrial practice because:

- The basic idea behind it is relatively simple,
- Very little or no modification is required to the basic formulation of MPC to handle multivariable plants, and

- It can be more powerful than proportional integral derivative (PID) control.

The potential of MPC has been realised by many industries such as the chemical, food processing, automotive and aerospace industries [104]. As a result its use has increased over the years because:

- It can easily incorporate actuator limitations.
- The plant is able to operate closer to the constraints.
- It is able to handle input constraints thereby never generating inputs that violate system limitations. As a result integrator wind-up is no longer an issue (integrator wind-up is present in conventional controllers when long duration set-point errors cause integrator outputs to exceed the saturation limits leading to larger overshoots and possible instability).
- It has the ability to control a great variety of processes including those with minimum phase, long time delay or open loop unstable characteristics.
- It is able to handle large complex systems with hundreds of controlled and manipulated variables.

The major disadvantage of MPC is the requirement of an accurate model. Another issue is that to handle constraints, predictive controllers require the online solution of quadratic programs which can be computationally very demanding. The computational burden of repeatedly solving optimisation problems often limits achievable sample rates to slower than desired. This can be a significant issue in aerospace systems where the sample rates are quite high. The leading theoretical challenges for MPC have been to guarantee feasibility and stability.

The performance success of MPC is highly dependent on the accuracy of the open loop predictions which are provided by the process model. Inaccuracies in the process model can often lead to predicted trajectories which differ from actual plant behaviour. This difference in the plant and model is referred to as plant-model mismatch or model uncertainty which can result in sluggish or unstable control performance. Some MPC controllers explicitly handle process model uncertainties when calculating the optimal control policies; these are termed robust predictive controllers. The general idea behind these controllers is similar to H_∞, where the worst case disturbance effect is minimised.

3.2.2 Nonlinear Model Predictive Control

There has been a recent rise in interest in nonlinear model predictive control (NMPC) within the control community mainly due to the increase in available computing power which is now capable of handling the computing demands of solving nonlinear optimization problems. The overall structure of NMPC is the same as that of linear MPC, in that the same receding horizon principle is adopted. The big advantage of NMPC is the incorporation of a nonlinear process model for highly-nonlinear systems. These nonlinear models are based on "first principles" and are obtained from an understanding of the physical nature of the system [90].

Linear MPC has been popular since the 70s whereas the interest in NMPC began in the 90s and has been driven by the fact that today's processes need to be operated under tighter performance specifications, with more and more constraints being imposed from environmental and safety considerations. More often than not, these demands can only be met when process nonlinearities and constraints are explicitly considered in the controller design [4]. The major limitation of linear MPC is that plant behaviour is described by a linear dynamic model making it unsuitable for both moderately as well as highly nonlinear processes which have large operating regimes [64]. Conceptually NMPC is similar to linear MPC. One of the reasons for the more frequent use of NMPC in the process industry is the time scales encountered which are in the order of minutes, making real-time requirements less severe than in aerospace applications.

The main characteristics of NMPC, many of which are shared by linear MPC, are:

- Prediction is based on non linear models.
- State and input constraints are explicitly handled.
- A specified performance criteria can be minimised on-line.
- In general, predicted behaviour is different from closed loop behaviour.
- The application of NMPC requires the online solution of an open loop optimal control problem.
- The system states must be measured or estimated to carry out the predictions.

NMPC has the potential to improve process operation, however, at the same time it offers theoretical and practical problems that are more challenging than those associated with its linear

counterpart. Most of these challenges are related to the nonlinear program which must be solved online at each sampling period. In addition, nonconvex nonlinear programs have several local minima, hence methods for obtaining the global minimum have to be applied (which increases the computational cost) [105].

One of the main issues in NMPC is ensuring the stability of the closed loop system from the utilisation of a finite horizon. The problem with a finite prediction and control horizon is that the predicted open-loop and the resulting closed-loop behaviours are in general different. The most obvious means of achieving stability is to use an infinite horizon, but the main drawback then is that at every sampling instance an infinite dimensional problem must be solved. An important characteristic of NMPC is that it possesses inherent robustness as it can deal with input model uncertainties without taking them directly into account. However, a problem with the standard NMPC setup is the open-loop nature of the control scheme, because essentially no feedback is used during the sampling periods.

The next section discusses optimal control techniques that can be used to solve NMPC problems.

3.2.2.1 Optimal Control Techniques

Nonlinear model predictive control involves finding the online solution of a receding horizon optimization problem. For the solution to be practically feasible the optimisation must be performed within the time constraints governed by the sampling period of the application [27]. A consequence of using a nonlinear dynamic model, as opposed to a linear dynamic model as done in MPC, is the need to solve a nonconvex nonlinear programming problem with a dramatic increase in complexity. Hence when designing and implementing NMPC strategies, consideration must be given to computational efficiency [27]. Long et. al [85] show that globally optimal NMPC methods can provide benefits over local techniques and can be successfully used for online control. As per Cannon [27], NMPC methods are generally based around:

1. Tailoring nonlinear programming algorithms to fit the structure of the online optimization.
2. Parametrising the predictions in terms of degrees of freedom. This directly affects the size of the online optimisation problem and in turn the computational burden of the NMPC strategy.

Cannon [27] provides an excellent review of all currently available computationally efficient

NMPC strategies, (described below), dividing them into two categories: development of optimisation algorithms or methods based on modifying the NMPC formulations. In my work I seek to modify the NMPC formulations.

NMPC strategies fall into the following types:

- Direct method optimisation. Here the aim is to apply nonlinear programming algorithms to make them fit within the NMPC receding horizon structure. In general a direct solution uses a finite parametrisation of the controls and/or constraints and the goal is to find an approximation of the original open-loop optimal control problem. The resulting finite dimensional optimization problem is solved using standard static optimization techniques that can be applied either sequentially or simultaneously.

  - Sequentially [27]: The model trajectories are predicted at each iteration, including methods based on successive linearisation, of the prediction model. In this method numerical integration is used in every iteration step of the optimization strategy to exactly solve the differential equations (or difference equations in the discrete time case). This means that the solution of the system dynamics is implicitly sought during the integration of the cost function and only the input vector at discrete points appears in the optimization problem as optimisation degrees of freedom.
  - Simultaneously [27]: The system states along with the controls are treated as optimization variables and the model dynamics form a part of the system constraints. Hence the system dynamics enter the optimisation as nonlinear constraints at each sampling point such that, at each such point the constraint $s = x(t, u)$ must be satisfied, where s is an additional degree of freedom in the optimization problem and is the initial condition for the sampling interval, t is time, u refers to the control inputs and x is a vector of states. One requirement of this constraint is that all the state trajectory pieces must fit together once the optimization has converged. The two most popular versions of this method are direct collocation and direct multiple shooting.

- Hamilton-Jacobi-Bellman (HJB) [4]: These optimal control methods do not utilise direct optimisation of predicted control trajectories used by conventional MPC solutions. Instead they search online for numerical solutions to optimal control formulations of the NMPC

receding horizon problem. The approach is based on the direct solution of the Hamilton-Jacobi-Bellmann partial differential equations (PDEs) [27]. Thus, rather than seeking the optimal $u(t)$ trajectory the problem is solved by finding the solution for all $x(t)$ where again $u(t)$ are the control inputs at time t and $x(t)$ are the states at time t. The solution is a state feedback law of the form $u^* = k(x)$ and is valid for all initial conditions. The main disadvantage of this method is the requirement of a complete solution, making it computationally demanding in addition to suffering from the curse of dimensionality, which means it can only be solved for small systems.

- Euler-Lagrange differential equations (DEs) / Calculus of Variations / Maximum Principle [27]: Again an explicit solution is sought of the input as a function of time $u(t)$ using calculus of variations. In this case the solution is not a feedback law, therefore it is only valid for the specified initial condition. This approach can be thought of as the application of the necessary conditions for constrained optimization with the optimization being infinite dimensional. The optimal control problem then becomes the seeking of a solution for a boundary value problem (BVP) where the approach is often to first optimise then discretise [37]. Due to the fact that an infinite dimensional problem must be solved this approach is not suitable for real-time applications.

- Modification or approximation of cost and constraints [27]: These methods are based on the modification or the approximation of the cost and constraints of the receding horizon optimization problem and involve the use of feasible sets (computed offline) for optimization variables in conjunction with cost approximations. Feasible sets are also used in combination with cost bounds corresponding to partitions of the states space which are determined offline.

- Reparameterisation [27]: The degrees of freedom in predictions are reparameterised to reduce the size of the receding horizon optimisation. One strategy uses the interpolation between feedback laws computed offline.

NMPC requires repeated computation of solutions to the optimal control problems on a finite prediction horizon in order to generate feedback controls for dynamical processes. The given optimal control problems are approximations to an infinite-horizon counterpart, hence the choice of the prediction horizon is critical [75]. To achieve closed-loop stability long horizons are preferable, however they are computationally expensive.

Schafer et. al [111] present an extended partially reduced sequential programming problem (SQP) method within the direct shooting framework to a thermally coupled distillation column. Schaffer et. al claim that there are two main reasons why NMPC is not popular for real time applications; firstly it requires a rigorous physical model of the process based on first principles which can be time consuming and expensive. Secondly NMPC requires successive online solutions of constrained nonlinear optimization problems. Schafer et. al reduce computational time for the optimisation problems with low degrees of freedom by introducing a new direct shooting method that diminishes the number of directions for which directional derivatives are evaluated with this number being independent of the state dimension.

Direct methods are normally applied for online solutions hence only direct methods will be investigated further.

3.2.2.2 Direct Methods

Direct methods involve reformulating the original infinite dimensional optimization problem as a finite nonlinear programming problem (NLP) by parameterising the controls and states (referred to as a simultaneous strategy). Typical parameterizations include collocation, finite differences or direct multiple shooting and this approach is based on the idea of discretise first, then optimise [37].

The NMPC applications investigated in this research all involve solving a BVP. The most popular direct methods are called shooting methods where the BVP is reduced to finding the solution to an initial value problem (IVP) by assuming initial values which would have been provided if the given ordinary differential equation (ODE) were an IVP. The calculated boundary value is compared to the real boundary value and based on a trial and error method (or some other scientific approach) with the aim being to come as close as possible to the boundary value [3]. In other words two boundary values must be satisfied in a two-point BVP ODE, but a BVP is much more difficult to solve than an IVP, because in a second order IVP $x(0)$ and $x'(0)$ are given but for a BVP only $x(0)$ and $x(a)$ (where a is the end point) are given. Another initial condition is required to solve the BVP, hence in shooting methods guesses are made for $x'(0)$ with the hope that the computed solution satisfies the second boundary condition. If the boundary condition cannot be satisfied another shot (or guess) is made [74]. Variations on this theme are described below:

3.2.2.2.1 Direct Single Shooting Direct Single Shooting is also known as control vector parameterization [62]. As previously mentioned, shooting methods are IVP solutions for solving BVPs with the concept being to take the more complicated BVP and convert it into a simpler IVP. Single shooting methods are the simplest to implement with the idea being to shoot at different angles until the boundary value is reached. The solution is achieved numerically via an iterative scheme. However single shooting methods cannot always be applied and the numerical results obtained are not always reliable. Furthermore numerical integration introduces discretisation errors leading to the major disadvantage of round off error accumulation which occurs when unstable IVPs have to be integrated.

3.2.2.2.2 Direct Multiple Shooting Direct multiple shooting, first introduced by Bock and Plitt [15] in 1983, is a fast off-line method for optimization problems in ODEs and differential algebraic equations (DAEs). It is a refinement of the single shooting method, with the horizon being divided into elements and each segment integrated separately. This method was introduced to reduce round off errors inherent in the single shooting method, which it achieves by subdividing the solution interval by a mesh with integrations being performed over each subinterval. That is, the IVPs are integrated numerically resulting in a two-level discretisation process; the first level consisting of a coarse mesh and the second a finer discretisation in each subinterval. The more the shooting points the lower the round off error, however there is a point of diminishing returns [9]. Multiple shooting is said to have better numerical properties compared to single shooting due to decoupling and state constraints at segment junctions. Advantages include [38]:

- The fact that it is a simultaneous strategy means suitable embedding techniques can be used to exploit the solution information previously obtained in controls and states and derivatives in subsequent optimization problems.
- State-of-the-art differential algebraic equation (DAE) solvers can be utilised to calculate the function values and derivatives quickly.
- Integrations are decoupled on different multiple shooting intervals making it suitable for parallel computation.
- Control and path constraints as well as boundary conditions are easily handled.

Multiple shooting methods are very popular for NMPC applications.

3.2.2.2.3 Collocation In 2010 Tamimi and Li [116] proposed a new algorithm for NMPC, a combination of the multiple shooting method and the collocation method, used to compute the function values and gradients in the NLP. In collocation methods a finite-dimensional space of candidate solutions is required to satisfy a differential equation exactly at the nodal points. The motivation behind collocation methods is that integrating differential equations is expensive (as is done in shooting methods). Also, sophisticated integrators are accurate but not necessarily consistent, and noisy derivatives means poor NLP convergence. In collocation methods IVPs are not explicitly integrated and approximate solution representations are sought over the prediction horizon. The basic idea entails discretising the controls AND the state variables, hence the problem is transcribed into discrete form in one step. Collocation methods can be implemented using a variety of numerical techniques such as the forward Euler method or more sophisticated one-step methods such as Runge-Kutta. The forward Euler method is conceptually simple and easy to implement but is numerically unstable and requires a small step length to achieve an accurate solution. Collocation methods can also make use of Lagrange Polynomials, which is equivalent to an implicit Runge-Kutta method with good numerical stability properties making it attractive for solving stiff (numerically unstable) systems. These techniques allow straightforward handling of state and control constraints.

In this section, linear and nonlinear model predictive control along with optimal control techniques used to implement MPC were detailed. In the next section the optimal control techniques necessary for the implementation of NMPC in this research are specifically discussed.

3.3 Numerical Methods: Discretisation

Numerical methods are an important component in the solution of optimal control problems. Since the problem at hand is often given in continuous time, it is necessary to discretise the problem for computer implementation and numerical analysis. The discretised solution will never provide the exact solution as it only gives the solution at discrete points and not over all time, leading to errors between the exact solution and the approximate solution provided by the discretised model. Thus, when choosing a discretisation method many factors need to be considered.

In this section a discussion of the discretisation methods commonly used in the solution of optimal control problems is presented with an emphasis on direct methods.

3.3.1 Problem Formulation

The path-constrained trajectory optimisation problem forms the essence of this research. An excellent exposition given by Williams [124] is summarised below. Note that the vector of unknown parameters, $\mathbf{p}$ in [124], has been omitted from this summary.

3.3.1.1 Primal Optimal Control Problem: Problem A

Generally speaking the aim of optimal control is to determine the state and control pair that minimises the cost functional J. That is, if a state and control pair is represented by $\{\mathbf{x}(t), \mathbf{u}(t)\}$ then the aim is to minimise:

$$J = \mathcal{M}\left[\mathbf{x}(t)\right] + \int_{t_0}^{t_f} \left[\mathcal{L}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)\right] \mathrm{d}t, \tag{3.1}$$

subject to the nonlinear state equations:

$$\dot{\mathbf{x}}\left(t\right) = f\left[\mathbf{x}\left(t\right), \mathbf{u}\left(t\right), t\right], \tag{3.2}$$

the initial and terminal constraints

$$\psi_0\left[\mathbf{x}\left(t_0\right)\right] = 0, \tag{3.3}$$

$$\psi_f\left[\mathbf{x}\left(t_f\right)\right] = 0, \tag{3.4}$$

the mixed state-control path constraints

$$\mathbf{g}_L \leq \mathbf{g}\left[\mathbf{x}\left(t\right), \mathbf{u}\left(t\right), t\right] \leq \mathbf{g}_U, \tag{3.5}$$

and the box constraints

$$\mathbf{x}_L \leq \mathbf{x}\left(t\right) \leq \mathbf{x}_U, \quad \mathbf{u}_L \leq \mathbf{u}\left(t\right) \leq \mathbf{u}_U, \tag{3.6}$$

where:

$\mathbf{x} \in \mathbb{R}^{u_x}$	are the state variables,
$\mathbf{u} \in \mathbb{R}^{u_u}$	the control inputs,
$t \in \mathbb{R}$	the time,
$\mathcal{M} : \mathbb{R}^{n_x} \times \mathbb{R} \to \mathbb{R}$	the terminal, non-integral cost, also known as Mayer component,
$\mathcal{L} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}$	the integral cost known as the Bolza component,
$\psi_0 \in \mathbb{R}^{n_x} \times \mathbb{R} \to \mathbb{R}^{n_0}$	the initial point conditions,
$\psi_f \in \mathbb{R}^{n_x} \times \mathbb{R} \to \mathbb{R}^{n_f}$	the final point conditions,
$g_L \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_g}$	the lower bounds on the path constraints, and
$g_U \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_g}$	the upper bounds on the path constraints.

Williams [124] explains that problem A exists in a primal space, where the optimal triplet $\{\mathbf{x}^*(t), \mathbf{u}^*(t), t_f^*\}$ (Here $*$ stands for optimal solution) is not easily found except for possibly trivial (zero) solutions.

Methods used to solve optimal control problems commonly fall under two categories; direct and indirect. Research in this area has focused mainly on direct rather than indirect methods as direct methods have better convergence properties than indirect methods and can be used quickly to solve a number of practical trajectory optimisation problems. A major drawback of certain indirect methods is having to derive the necessary conditions for optimality using Pontryagin's maximum principle (PMP) [124]. In addition many indirect methods require *a priori* information of the switching structure of the constrained and unconstrained subarcs in problems with state inequalities, a problem that has been addressed by utilising Homotopy methods to estimate the switching structure. This, however, leads to a significant increase in computational time. Direct methods, on the other hand, do not require derivation of the necessary conditions or estimates of the switching structure of the costates and the region of convergence is greatly increased compared to indirect methods. Direct methods solve Problem A *directly* by applying a discretisation process and using standard algorithms to solve the resulting optimisation problem. The direct methods have many advantages over indirect methods [124], especially with regards to the research questions under consideration in this thesis, and these advantages are detailed below in subsection 3.3.2.

For this research, it is necessary for the continuous time problem, Problem A, to be converted to a finite dimensional problem before attempting to solve it.

3.3.2 Direct Methods - Numerical Techniques

A multitude of discretisation techniques exist for converting the infinite dimensional problem to a finite dimensional one. In direct methods the mathematical programming problem, equations (3.1) to (3.6), is solved by considering either discretised inputs, or a combination of discretised inputs and states, as decision variables. The most common direct methods in practice are where control and state parameters are used as optimisation parameters. They can be further separated into the following two categories:

1. Local - Also known as direct collocation or direct transcription methods. Here a number of node points with arbitrary spacing are defined where both the state and control vectors are collocated. The state equations are enforced as equality constraints at internal collocation points between the nodes via implicit integration methods such as Simpson's rule or by Gauss-Lobatto (GL) quadrature rules. In other words the state equations are enforced locally.

2. Global - Also known as Pseudospectral Methods. For this approach globally orthogonal interpolating polynomials are utilised to approximate the state and control variables. Discretisation is based on the GL points for Legendre or Chebyshev polynomials. The state equations are enforced by differentiating the approximating polynomial at the corresponding GL points rather than through numerical integration. The GL points are the zeros of the derivative of the interpolating polynomial. Global methods are generally more accurate than local methods because the discrete adjoint multipliers retain the same order of accuracy as the state equations. This is not the case for certain classes of local methods, for example when Hermite-Simpson and particularly Runge-Kutta discretisations are employed.

The ease and speed with which difficult problems can be solved make direct methods highly appealing. The most popular discretisation methods include Hermite-Simpson and Legendre Pseudospectral which utilise different choices of basis functions to be used as decision variables in the optimisation. Such a selection determines different discretisations, that is node points.

When deciding on a discretisation process many important factors need to be considered as have been identified by Williams [124]:

- How accurate is the solution for a particular discretisation method given a number of optimisation variables?
- What is the computational expense of a particular discretisation method?
- How robust is a discretisation method to the initial guess?
- How are the answers to the previous questions influenced by problem complexity and the types of problems being considered?
- How are the answers to the previous questions affected by the choice of NLP solvers?

Williams [124] explains that the choice of the NLP solver affects the speed and robustness of solutions obtained using particular discretisations. For example a dense solver can result in significant increases in CPU time since, in this case, the constraint Jacobian and Hessian are treated as dense matrices and this increases the number of optimisation/decision variables.

The preceding information was a general overview of the area of discretisation. The area of numerical methods for a given application is an extensive research topic by itself. However, since the aim of this research is to explore fault tolerant control schemes, in the following section only the particular methods chosen for further investigation will be explored.

3.4 Implementation of Optimal Control Techniques

The previous section presented a discussion on the various discretisation techniques used to solve optimal control problems. The aim of this section is to investigate a few of these optimal control techniques in greater detail as well as to implement one of these methods in the situations encountered in this research on a 2-D robot model, to gain a thorough understanding of the fundamental workings of NMPC.

As mentioned previously in section 3.3.2, direct methods used to solve optimal control problems are classified according to the unknown parameters used in the optimisation. For this research one method from each of the categories listed in section 3.2.2.2 will be considered. There are many methods that fall under each category, and since bounds need to be placed on the research

being conducted, I have chosen only one from each of the categories in order to understand how they can be used to implement NMPC.

3.4.1 Optimisation Parameters: Controls Only - Direct Single Shooting

Shooting methods solve a two point BVP, for example:

$$y'' = f(x, y, y'), \quad y(a) = y_a, \quad y(b) = y_b. \tag{3.7}$$

A BVP is an ODE where information is given on the boundaries, a and b. Numerical methods designed for solving ODEs, for example, Euler's method and the Runge Kutta method, are generally used to solve IVPs. However, to solve an IVP a sufficient number of initial conditions must be provided, which for equation (3.7) includes the derivative at the initial boundary. Shooting methods convert two point BVPs into IVPs by guessing the value of the derivative at the initial boundary. Every time a guess is made a "shot" is fired in an attempt to hit the end boundary. This "shot" involves using differential equation solvers to find a solution. It is an iterative process where "shots" are made until the end boundary is reached to within a desired tolerance.

Direct single shooting optimal control takes this concept and evaluates the optimal inputs for any given control system. In direct single shooting optimal control, only the controls are used as optimisation parameters with optimisation being performed from an initial time, t_0, to the final time t_f. The time interval is divided into N equally spaced intervals. On this grid the controls, $u(t)$, are discretised and are piecewise constant, $u(t) = q_i$ where $i = 0, 1, \ldots, N$, and the following initial value problem is solved:

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t, \mathbf{q})), \quad t \in [t_0, t_f]. \tag{3.8}$$

A nonlinear program of the following form is constructed by including the cost function equation (3.1) on the decision variables x and u:

$$\min_q \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}(t, \mathbf{q}), \mathbf{u}(t, \mathbf{q}), t) \; \mathrm{d}t + \mathcal{M}(\mathbf{x}(t_f, \mathbf{q})), \tag{3.9}$$

subject to:

$$h(\mathbf{x}(t_i, \mathbf{q}), \mathbf{u}(t_i, \mathbf{q})) \geq 0, \qquad i = 0, \ldots, N \qquad \text{(discretised path constraints)}, \tag{3.10}$$

$$r(\mathbf{x}(t_f, \mathbf{q})) = 0, \qquad \text{(terminal constraints)}. \tag{3.11}$$

The optimisation procedure calculates the optimal control inputs which minimise the cost function given in (3.9).

The advantages of this method are:

- Fully adaptive, error controlled state-of-the-art ODE or DAE solvers can be utilised.
- It has only a few optimisation degrees of freedom even for large ODE or DAE systems.
- Only initial guesses for the control degrees of freedom are needed.
- There is simplicity of design and implementation.

However there are weaknesses in this method:

- Knowledge of the state trajectory, x, cannot be used in the initialisation.
- Unstable systems are difficult to handle.
- The ODE solution $x(t, q)$ can depend very non-linearly on q.

In regards to NMPC N equally spaced discretisations points are placed along the prediction horizon length, H_p. A full nonlinear process model is used by the optimal controller to integrate the states between each of the discretisation points.

Figure 3.2 shows an illustrative description of the Direct Single Shooting Method.

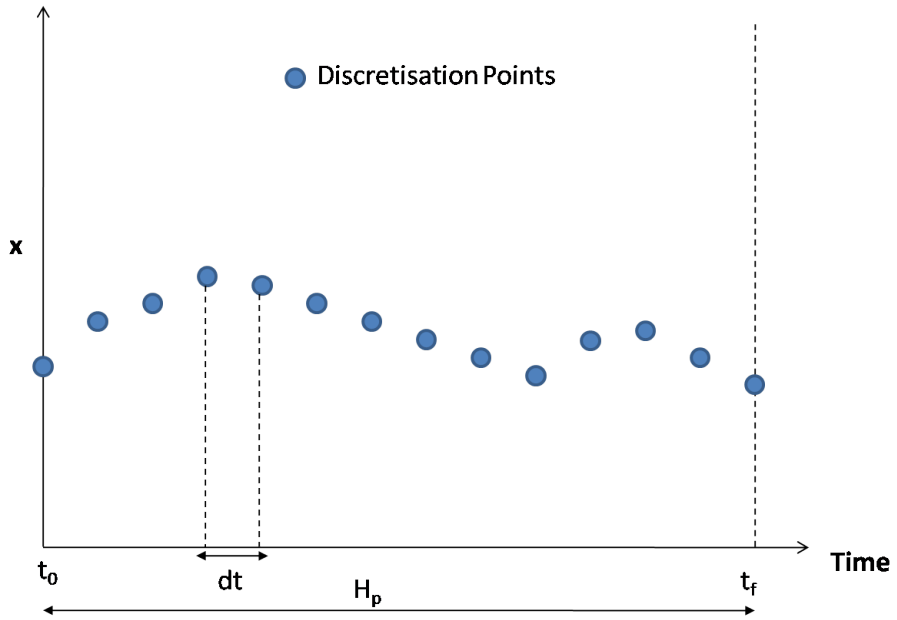

Figure 3.2: Direct Single Shooting - Diagram of the division of the time interval in the single shooting method.

3.4.2 Optimisation Parameters: Controls and Some States - Direct Multiple Shooting

Direct Multiple Shooting is based on the same idea as that of Single Shooting, i.e. converting a BVP to an IVP and "shooting" from the initial boundary until the end boundary is reached. The main difference is that, rather than converting a BVP to a single IVP, the problem is converted into multiple IVPs. This is achieved by dividing the interval of computation, $[t_0, t_f]$, into M subintervals and solving an IVP over each subinterval. All of the solutions over the subintervals can be pieced together to form a continuous trajectory/solution, if and only if the solutions to the IVPs match at the beginning and end of each subinterval, referred to as matching conditions. These matching conditions introduce algebraic equations which must be satisfied along with the boundary conditions.

The optimisation parameters in this case are the controls as well as the states at the beginning and end of each interval. The controls are discretised in each subinterval over N_u points, hence $u(t) = q_i$ where $i = 0, 1, \ldots N_u$. An ODE solver is used to integrate the states over each subinterval. Let $x_j(t_j) = s_j$ represent the solution at the beginning and end of each subinterval, where $j = 0, 1, \ldots M$. Then:

$$\dot{x}_{i,j} = f(x_{i,j}(t_i), q_{i,j}), \quad t \in [t_i, t_j], \tag{3.12}$$

$$x_i(t_j) = s_j, \tag{3.13}$$

where the $\mathbf{x}_{i,j}$ is the solution at the i^{th} node in the j^{th} subinterval.

The resulting NLP is the same as that given in the direct single shooting method except with extra continuity/matching constraints:

$$\min_{s,q} \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}(t,q), \mathbf{u}(t,q), t) \ \mathrm{d}t + \mathcal{M}(x(t_f, q)), \tag{3.14}$$

subject to:

$$s_0 - x_0 = 0, \qquad \text{(initial value)} \tag{3.15}$$

$$s_j - x_{t_j, s_j, q_j} = 0, \qquad j = 0, \ldots, M \qquad \text{(continuity)} \tag{3.16}$$

$$h(s_j, q_j) \geq 0, \qquad i = 0, \ldots, N-1 \qquad \text{(discretised path constraints)} \tag{3.17}$$

$$r(s_N) = 0. \qquad \text{(terminal constraints)} \tag{3.18}$$

Advantages of Direct Multiple Shooting include:

- Knowledge of some of the states is included in the optimisation.
- It can robustly handle unstable systems and path state and terminal constraints.
- It is able to exploit advantages of using ODE and DAE solvers.

In regards to NMPC the prediction window is divided into smaller subintervals and the full non-linear process model is used for integration between discretisation points in every subinterval.

Figure 3.3 shows an illustrative description of the Direct multiple shooting method.

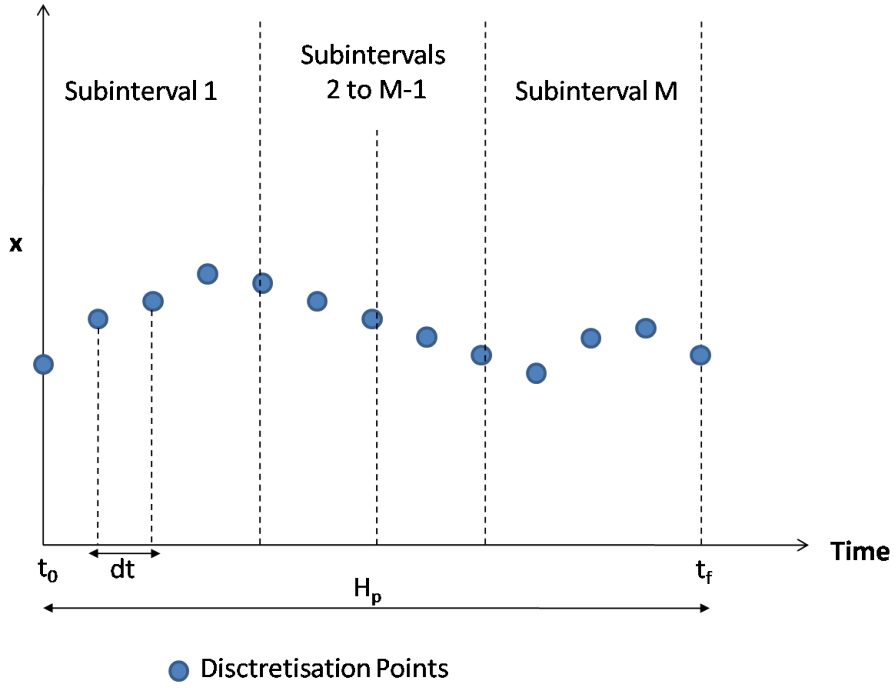

Figure 3.3: Direct Multiple Shooting - Diagram of the division of the time interval in the multiple shooting method

3.4.3 Optimisation Parameters: Controls and All States - Direct Transcription

Direct Transcription involves fully discretising the problem (all controls and all states) and then solving the discrete problem numerically. Numerous direct transcription methods exist for solving optimal control problems differing only in the way in which they approximate the state equations. The discretisation method used to approximate the state equations must be combined with a method for approximating the integral in the generalised Bolza problem and such discretisation techniques are either integration or differentiation based. Pseudospectral methods, for example, are all differentiation based methods; they rely on differentiating Lagrange Polynomial expansions of the approximating polynomials for the states. Hermite-Simpson based techniques are often thought of as integration methods. A comparison of various methods has been given in [124]. For this research an integration based method as well as a differentiation based method will be investigated, hence these two methods are described in detail in the next couple of sub-sections.

3.4.3.1 Integration Based Methods (Euler)

Both the controls and the states are discretised on a fine grid. The controls are selected to be piecewise constant with values q_i, where $i = 0, 1, \ldots, N$, at each point on the interval $[t_0, t_f]$ and the states are denoted by $s_i = x(t_i)$ at each of the grid points along the time interval. In the collocation method the infinite dimensional ODE:

$$\dot{x}(t) - f(x(t), u(t)) = 0, \quad t \in [t_0, t_f] \tag{3.19}$$

is replaced by a finite number of equality constraints:

$$c_i(q_i, s_i, s_{i+1}) = 0, \quad i = 0, 1, \ldots, N-1, \tag{3.20}$$

where

$$c_i(q_i, s_i, s_{i+1}) = \frac{s_{i+1} - s_i}{t_{i+1} - t_i} - f(s_i, q_i). \tag{3.21}$$

The integrals on the collocation intervals are approximated by:

$$l_i(q_i, s_i, s_{i+1}) \approx \int_{t_i}^{t_{i+1}} L(x(t), u(t)) \, \mathrm{d}t = L(s_i, q_i)(t_{i+1} - t_i). \tag{3.22}$$

After discretisation the following sparse NLP is obtained:

$$\min_{s,q} \sum_{i=0}^{N-1} l_i(q_i, s_i, s_{i+1}) + \mathcal{M}(s_N), \tag{3.23}$$

subject to:

$$s_0 - x_0 = 0, \qquad \text{(fixed inital value)} \tag{3.24}$$

$$c_i(q_i, s_i, s_{i+1}) = 0, \quad i = 0, \ldots, N-1, \qquad \text{(discretised ODE model)} \tag{3.25}$$

$$h(s_i, q_i) \geq 0, \quad i = 0, \ldots, N, \qquad \text{(discretised path constraints)} \tag{3.26}$$

$$r(s_N) = 0. \qquad \text{(terminal constraints)} \tag{3.27}$$

Advantages of collocation are:

- Knowledge of the state trajectory, x, can be used in the initialisation.
- A sparse NLP is obtained.
- Fast local convergence is possible.

- Can treat unstable systems well.
- Can easily cope with state and terminal constraints.

A major disadvantage of the collocation method is that adaptive discretisation error control needs regridding and thus changes the NLP dimensions. Hence collocation often does not address the question of proper discretisation error control.

In regards to NMPC with direct transcription the prediction horizon is divided into N discretisation or collocation points. The optimisation vector comprises the states and controls at ALL node points. Points are equally spaced in the same fashion as that given by figure 3.2 with $\triangle t$ the time between collocation points.

3.4.3.2 Derivative Based Methods

Derivative based methods, also known as pseudospectral methods, differ from many traditional discretisation methods. The difference lies in that the focus is shifted from the DE to the tangent bundle, which comprises all tangent vectors, together with information of the point at which they are tangent. For this reason they can be seen to resemble finite element methods; however they provide a better convergence rate. The underlying idea is to represent the solution f via a truncated series expansion and to use analytic differentiation of the series to obtain spatial derivatives of f. The spectral differentiation matrix, $\mathcal{D}_N$, is a linear mapping of a vector of N function values $\{f(x_i)\}$ to a vector of N derivative values $\{f'(x_i)\}$. The calculation of $\mathcal{D}_N$ is dependent on the choice of the approximating series and the location of the points $\{x_i\}$. An example is the use of the discrete Fourier transform with equally spaced nodes for analysis purposes, which is based on the idea of using (sums of) periodic functions to approximate given functions. A Chebyshev or Legendre series is used for bounded domains where the derivatives are calculated at Chebyshev or Legendre nodes or extreme points. If f is infinitely differentiable then the remainder of the truncated series will go to zero super-algebraically; that is, it will decrease faster than any finite power of $\frac{1}{N}$ [57] with the rate restricted only by the global smoothness of the function; this is referred to as spectral accuracy [58]. Hence pseudospectral methods offer a very good convergence rate (spectral accuracy) and for smooth problems spectral accuracy means an exponential convergence rate [107].

One advantage pseudospectral methods have over finite element or finite difference methods is that the underlying polynomial space is spanned by orthogonal polynomials that are infinitely

differentiable global functions. Traditional pseudospectral methods use global Lagrange interpolating polynomials to expand the state and control trajectories. These polynomials are based on orthogonal polynomials from the Jacobi family such as Legendre or Chebyshev polynomials, which are orthogonal with respect to a specific weight function over a fixed interval. The idea of expanding the state and control variables in terms of Lagrange interpolating polynomials at suitably chosen points can be seen in the work of Elnagar et. al [43], Fahroo and Ross [45] and Ross and Fahroo [107], where pseudospectral methods are used for solving fluid dynamics problems.

Generally speaking there are two ways of constructing a polynomial approximation to the output solution $y(t)$:

1. Use an interpolating polynomial between the values $y(t_j)$ at node points t_j, or
2. Use a series expansion in terms of orthogonal polynomials.

Most commonly the nodes chosen for approximating the optimal control problem in both of the cases given above are Legendre Gauss Lobatto (LGL) points and, together with the properties of the Lagrange polynomials, the state equations and the state and control constraints, can easily be transformed in to algebraic equations in terms of values of the state and control variables at the nodes. The constraints are imposed as differential constraints at the LGL points via a differentiation matrix. As Fahroo and Ross [45] explain, an arbitrary choice of node points can lead to very poor results in interpolation of a function. Hence Gauss quadrature points (such as LGL points) are chosen to give the best accuracy and then the derivatives of the interpolating polynomials at these node points are given exactly by a differentiating matrix. LGL points minimise the L_2 norm of the approximation error. One of the features of the LGL points is that the nodes cluster around the end points. The node points obtained via Gauss quadrature cluster around the endpoints of the interval resulting in the avoidance of the Runge phenomenon. The Runge phenomenon is a term given to the divergence of the approximating solution at the end points of interpolation on equispaced interpolation points.

The choice of collocation points is crucial in pseudospectral methods. The three most common sets of collocation points are the Legendre-Gauss, Legendre-Gauss-Radau and Legendre-Gauss-Lobatto. These three sets are the roots of a Legendre polynomial and/or linear combinations of Legendre polynomials and their derivatives. The three sets are defined on the domain $[-1, 1]$ and

differ according to which end points are included. The Legendre-Gauss points do not include either end points, Legendre-Gauss-Radau include only the endpoint -1, and the Legendre-Gauss-Lobatto points include both end points. Ross and Karpenko [108] state that Gauss-Radau is best used for an infinite horizon problem whereas Gauss-Lobatto is more suited for finite horizon problems.

Another popular choice for collocation points are Chebyshev-Gauss-Lobatto points; these are based on the roots of the derivatives of the Chebyshev polynomials [45]. Chebyshev and Legendre polynomials are both from the Jacobi family of polynomials. For approximating BVPs the Gauss-Lobatto points are a logical choice due to the combination of high interpolation accuracy and high accuracy in quadrature approximations. LGL points minimise the L_2-norm of the approximation error whereas the Chebyshev-Gauss-Lobatto (CGL) points minimise the max-norm of the approximation error. Williams [125] states that the LGL points provide maximum accuracy for quadrature approximations whilst, at the same time, avoiding the Runge phenomenon during interpolation.

While traditionally pseudospectral methods did utilise Lagrange interpolating polynomials, Gauss-Lobatto quadrature rules with Jacobi polynomials as the interpolant have been used more frequently. Jacobi polynomials are orthogonal over the interval $(-1, 1)$ with respect to the weight function, $w(\tau) = (1-\tau)^{\alpha}(1+\tau)^{\beta}$. The Legendre polynomials are obtained when $\alpha = \beta = 0$ and, when $\alpha = \beta = -0.5$ the Chebyshev polynomials are produced, hence by varying α and β a different set of Jacobi polynomials are produced. Williams [123] generalises the pseudospectral method to consider collocation based on the roots of the derivatives of general Jacobi polynomials. Williams clarifies that the Legendre and Chebyshev are particular cases of the more general formulation and finds that by varying α and β there is a significant impact on computation time.

A pseudospectral method based on non classical orthogonal and weighted interpolating polynomials is presented in a more recent paper by Williams [126]. Here the author highlights the fact that the location of collocation points in traditional pseudospectral methods are more or less fixed. The method presented by Williams [126] generalises the existing methods to allow a much more flexible selection of grid points via an arbitrary selection of the orthogonal weight function and interval. This freedom of choice leads to a greater range of collocation points and differentiation matrices which in turn affects computation time. Over the years, the use of LGL

points for optimal control, has been seen as the most natural choice for Quadrature points, the main reason being their derivation on the basis of a unit weight function, which provides the highest accuracy for polynomial integrands.

Pseudospectral methods are collocation based methods, hence the optimisation parameter vector comprises the states and controls at ALL nodes. One of the main differences is the selection of node points that are not equally spaced. The nodes are carefully selected so as to avoid the Runge phenomena, to increase accuracy (providing spectral accuracy) and, to avoid numerical ill conditioning. As mentioned previously the most commonly used node points are the Chebyshev Gauss Lobatto points or the Legendre Gauss Lobatto points. For this work the Legendre Gauss Lobatto (LGL) points were chosen for implementation. The nodal points τ_k, $k = 0, \ldots, N$, lie in the interval $[-1, 1]$ where the endpoints are included. These node points are the zeros of the derivatives of the Legendre polynomials. A linear transformation is needed to map the computational domain, $\tau \in [-1, 1]$, to the physical domain, $t \in [t_0, t_f]$:

$$t = \frac{(t_f - t_0)\,\tau}{2} + \frac{(t_f + t_0)}{2}. \tag{3.28}$$

The central idea behind pseudospectral methods is the approximation of the states and controls with Lagrange interpolating polynomials of degree N such that:

$$\mathbf{x}_N(\tau) = \sum_{j=0}^{N} \hat{x}_j \phi_j(\tau_j), \tag{3.29}$$

$$\mathbf{u}_N(\tau) = \sum_{j=0}^{N} \hat{u}_j \phi_j(\tau_j), \tag{3.30}$$

where $\hat{x}_j(\tau_j)$ and $\hat{u}_j(\tau_j)$ are the coefficients of the interpolating polynomial. It is necessary that the coefficients $\hat{x}_j = \mathbf{x}(\tau_j)$ and $\hat{u}_j = \mathbf{u}(\tau_j)$, hence $\phi_j(\tau)$, are the Lagrange interpolating polynomials that interpolate the states and controls at the LGL points, given by:

$$\phi_j(\tau) = \frac{\left(\tau^2 - 1\right)\dot{L}(\tau)}{(\tau - \tau_j)\,N(N+1)\,L_N(\tau_j)}, \tag{3.31}$$

where L_N represents the Nth degree Legendre polynomial. This leads to the following NLP problem:

Minimise the cost function, J:

$$J_N = \mathcal{M}\left[x_N\right] + \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \mathcal{L}\left[\mathbf{x}_j, \mathbf{u}_j, \mathbf{t}_j\right] w_j, \tag{3.32}$$

subject to:

$$\mathbf{x}_0 - \mathbf{x}(t_0) = 0, \qquad \text{Initial Condition} \tag{3.33}$$

$$\frac{t_f - t_0}{2} \mathbf{D}_{j,k}\, \mathbf{x}(j) - \dot{\mathbf{x}}(j) = 0, \qquad \text{State Equations} \tag{3.34}$$

$$x_L \leq \mathbf{x} \leq x_U, \qquad \text{Upper and Lower bounds on states} \tag{3.35}$$

$$u_L \leq \mathbf{u} \leq u_U, \qquad \text{Upper and Lower bounds on control} \tag{3.36}$$

where w_j are the LGL weights given by:

$$w_j = \frac{2}{N\left(N+1\right)} \frac{1}{\left[L_N\left(\tau_j\right)\right]^2}, \quad j = 0, \ldots, N, \tag{3.37}$$

and $D_{j,k}$ is a differentiation matrix of size $(N+1) \times (N+1)$ the entries of which are given by:

$$D_{j,k} = \begin{cases} \frac{L_N(\tau_j)}{L_N(\tau_k)} \frac{1}{(\tau_j - \tau_k)}, & \text{if } j \neq k, \\ -\frac{N(N+1)}{4}, & \text{if } j = k = 0, \\ \frac{N(N+1)}{4}, & \text{if } j = k = N, \\ 0 & \text{otherwise.} \end{cases} \tag{3.38}$$

It can be seen from the above that the pseudospectral method uses a Gauss-Lobatto quadrature rule to approximate the integral (or the Bolza component) in the performance index. The state derivatives are approximated via analytical differentiation of the interpolating polynomials of the states.

3.4.3.3 NLP Solvers

Each of the methods stated above produce an NLP which must be solved at every time step across the prediction horizon, H_p. Once the optimisation is solved at the current time, t_j, only the first calculated control input is applied to the plant and the window is shifted by the sampling time step, t_s and the whole procedure begins again. Hence an NLP solver is required and is an integral part of NMPC. Williams [124] points that the different discretisation methods

are affected by the choice of NLP solvers in terms of speed and robustness of the solution obtained. Williams [124] provides an example where if a dense solver is utilised, i.e. one where the constraint Jacobian and Hessian are treated as dense matrices, then it can lead to significant increases in CPU time due to the increase in the number of optimisation variables. For this work SNOPT [54] is the solver of choice due to its popularity and because it is readily available. SNOPT solves the quadratic programming subproblem with a quasi-Newton approximation to the Hessian, via a large-scale sparse sequential quadratic programming algorithm. Please note that there are other, more efficient solvers which could be used [124] but are not used in this research because of the difficulty in gaining access to them.

In the next subsection, the optimal control methods given above, Direct Single Shooting, Direct Multiple Shooting and Direct Transcription methods are implemented to solve the well known Brachistochrone problem [40]. Based on the findings one of these methods is chosen for implementing NMPC in this research.

3.4.4 Brachistochrone

The Brachistochrone problem is a well known nonlinear problem. It is chosen because it is a nontrivial problem with an analytic solution and is very similar to the 2D robot problem that is to be addressed later. The analytical solution allows accuracy of the implementation of each method to be determined and is used as a benchmark in choosing a method to continue this research. The Brachistochrone problem, simply stated, is to find the shape of a wire such that a bead sliding on the wire without friction, in uniform gravity, will reach a given horizontal displacement in minimum time.

The analytical solution is given by:

$$x_b = \frac{g}{\omega^2}\left(\omega\, t - \sin\omega\, t\right), \tag{3.39}$$

$$y_b = \frac{g}{\omega^2}\left(1 - \cos\omega\, t\right), \tag{3.40}$$

where:

$$\omega = \sqrt{\frac{\pi\, g}{x_f}}, \tag{3.41}$$

x_b and y_b are the displacement values of the bead in the xy-plane, g is the gravitational force and x_f is the final x-displacement.

The optimal control problem is to minimise the cost function, J:

$$J = t_f, \tag{3.42}$$

subject to the equations of motion of the bead:

$$\begin{aligned} \dot{x} &= V \sin\theta, && (3.43)\\ \dot{y} &= V \cos\theta, && (3.44)\\ \dot{V} &= g \cos\theta, && (3.45) \end{aligned}$$

and the initial and terminal constraints:

$$\begin{aligned} x(0) &= 0, && (3.46)\\ y(0) &= 0, && (3.47)\\ V(0) &= 0, && (3.48)\\ x(t_f) &= x_f. && (3.49)\\ & && (3.50) \end{aligned}$$

Here t_f is the time taken to reach x_f and V is the speed of the bead.

The number of discretisation points was varied for each method to investigate their effect and to determine the method most suitable for developing the fault tolerant controller. The value of x_f is set to $0.5m$ and the value of g for this work is $1m/s^2$.

For the direct single shooing method the control points were chosen to be:

$$N_u = [5,\ 10,\ 25,\ 50,\ 150,\ 250,\ 500],$$

and for each control point the state points were varied:

$$N_x = [10,\ 20,\ 50,\ 100,\ 300,\ 500,\ 1000].$$

Similarly the number of sections for the direct multiple shooting method were set to:

$$M = [2,\ 5,\ 10,\ 20,\ 30],$$

and the control points chosen for each section were:

$$N_u = [2,\ 5,\ 10,\ 20,\ 50].$$

The coincidence points for both the collocation methods (Euler integration and Pseudospectral) were set to:

$$N = [5,\ 10,\ 50,\ 100,\ 300,\ 500,\ 800].$$

3.4.4.1 Accuracy of Solution Methods

I now assess the accuracy of each of the numerical methods by comparing the optimal solution produced by the numerical method with the analytical solution given in equation (3.39). For each optimal control method the number of discretisation points is varied to assess the affect of that level of discretisation on the solution.

3.4.4.1.1 Direct Single Shooting Results

The plots given in figures 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 and 3.10 show the solutions produced by the direct single shooting method by varying N_u and N_x. The plots given are of the bead trajectory ie. x vs y plots. The analytical solution is given in red and the numerical solution produced by the direct single shooting method is given in blue. All the plots show that for a given N_u the difference between the numerical solution and the analytical solution decreases as N_x increases. This is further exemplified by the individual error plots in the displacement in the x (figures 3.11, 3.12, 3.13, 3.14, 3.15, 3.16 and 3.17) and y (figures 3.18, 3.19, 3.20, 3.21, 3.22, 3.23 and 3.24) directions. The errors in these plots represent the magnitude of the displacement errors in x and y between the analytical and numerical solutions. The results show that five control points, $N_u = 5$ are not enough to produce a smooth accurate solution. A minimum of ten control points is required to produce a smooth solution. The results also show that there should be at least twice as many state points N_x as control points to minimise the error.

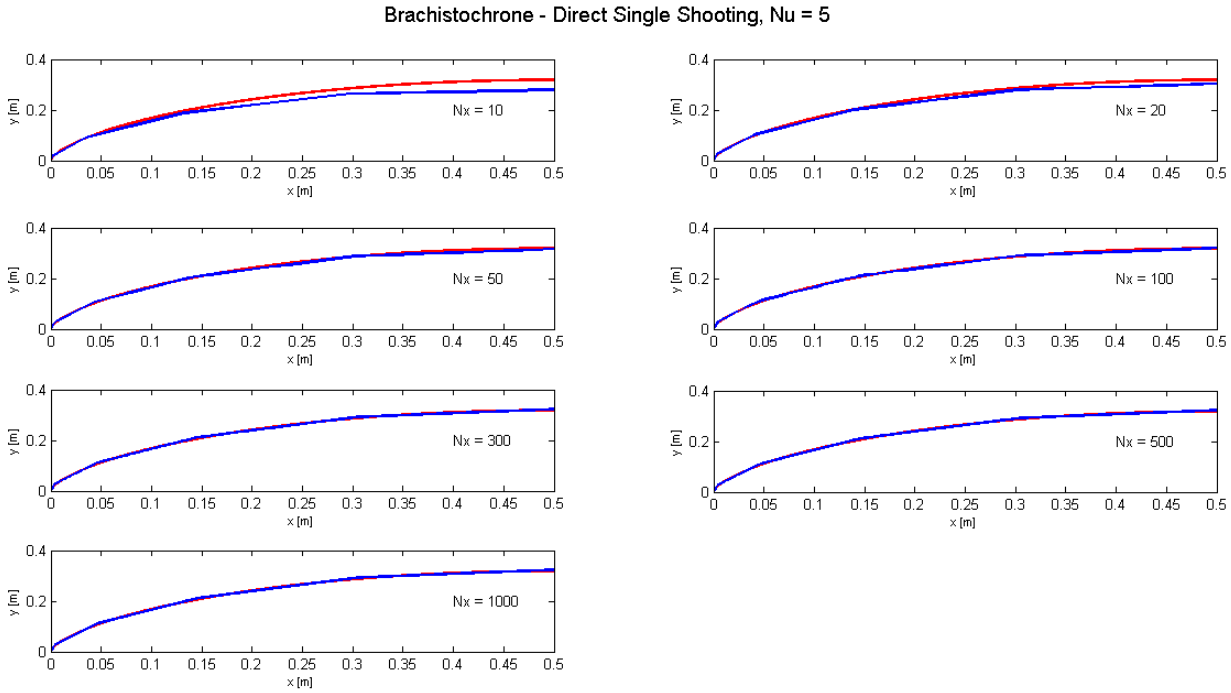

Figure 3.4: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 5$: Analytical Solution (Red) and Numerical Solution (Blue).

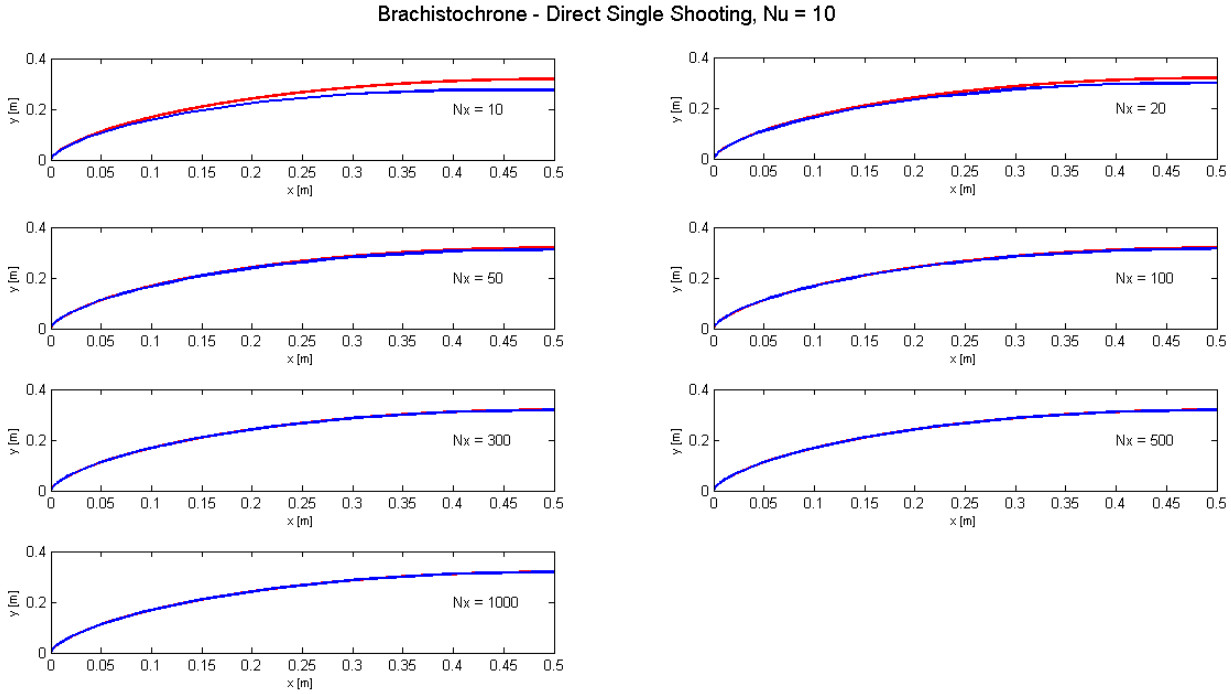

Figure 3.5: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 10$: Analytical Solution (Red) and Numerical Solution (Blue).

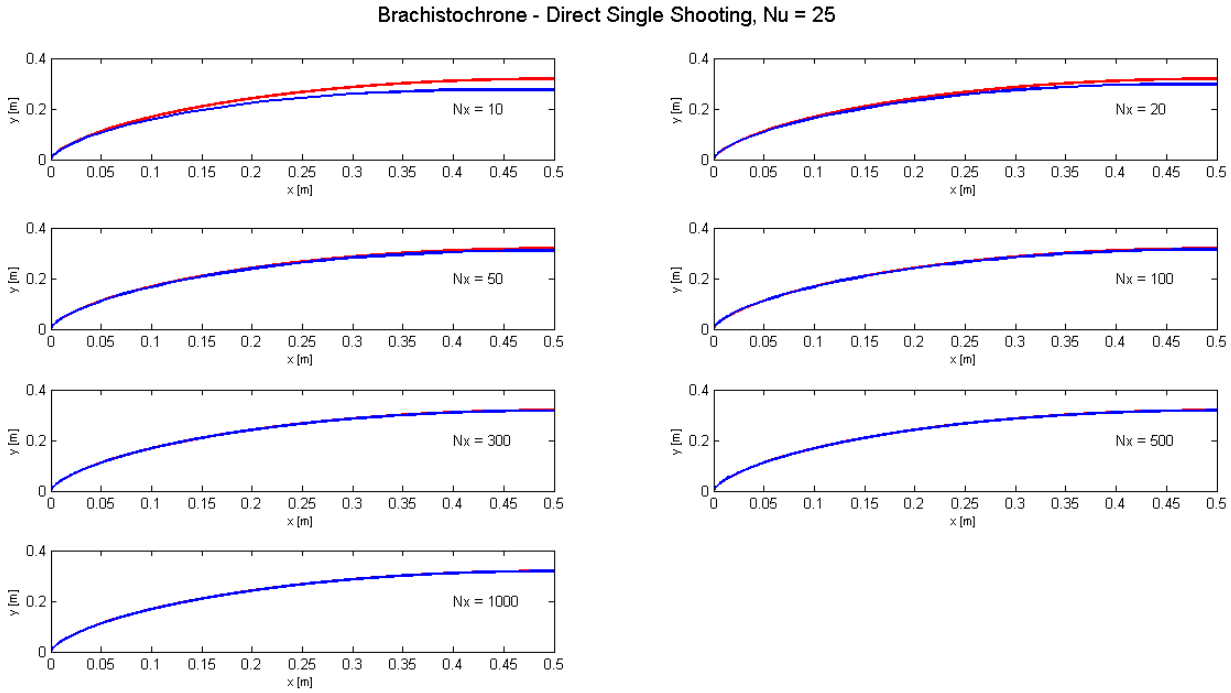

Figure 3.6: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 25$: Analytical Solution (Red) and Numerical Solution (Blue).

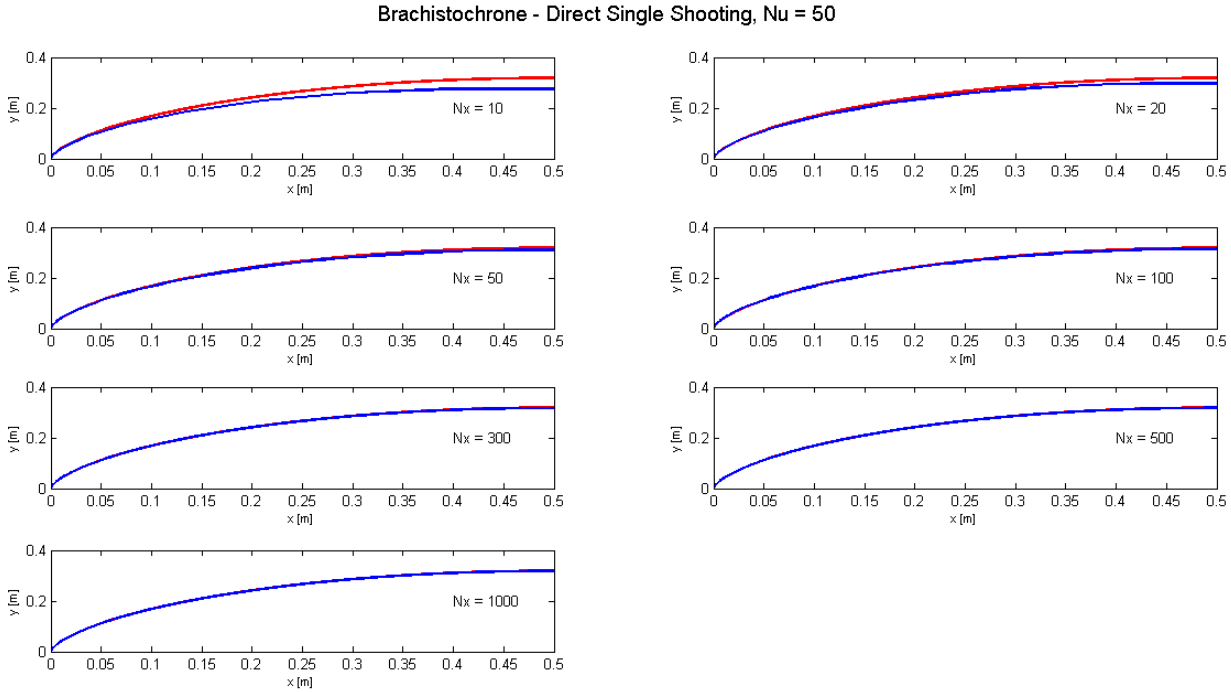

Figure 3.7: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 50$: Analytical Solution (Red) and Numerical Solution (Blue).

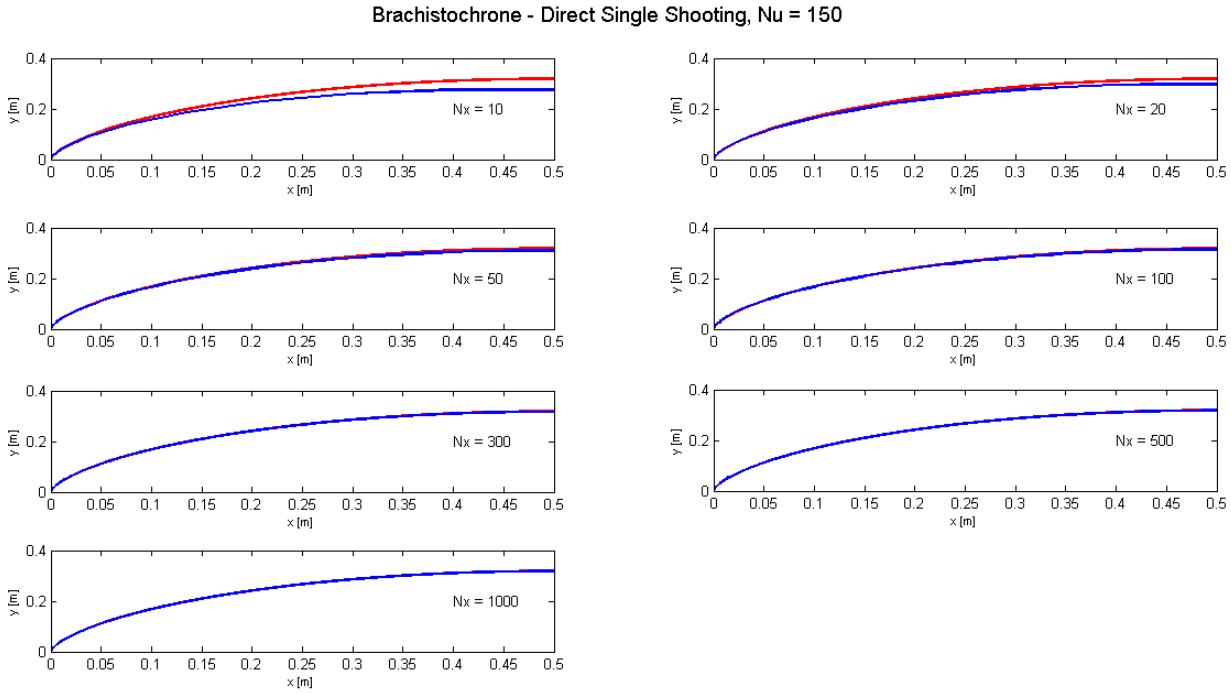

Figure 3.8: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 150$: Analytical Solution (Red) and Numerical Solution (Blue).

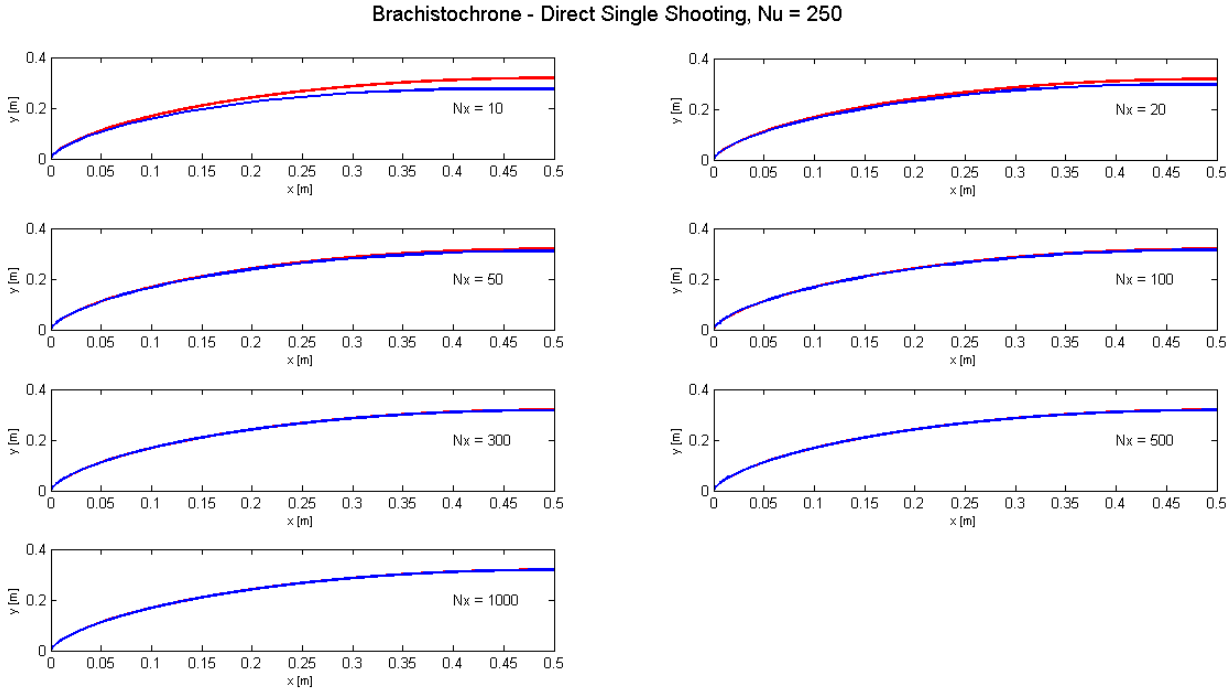

Figure 3.9: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 250$: Analytical Solution (Red) and Numerical Solution (Blue).

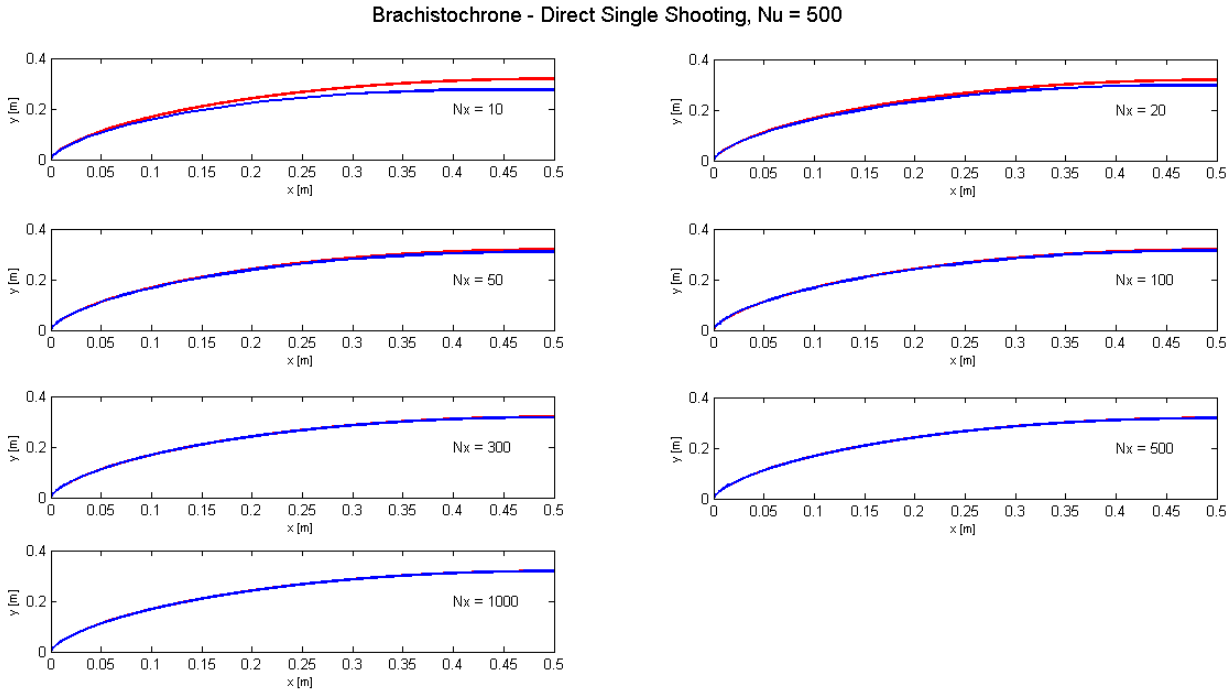

Figure 3.10: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $N_u = 500$: Analytical Solution (Red) and Numerical Solution (Blue).

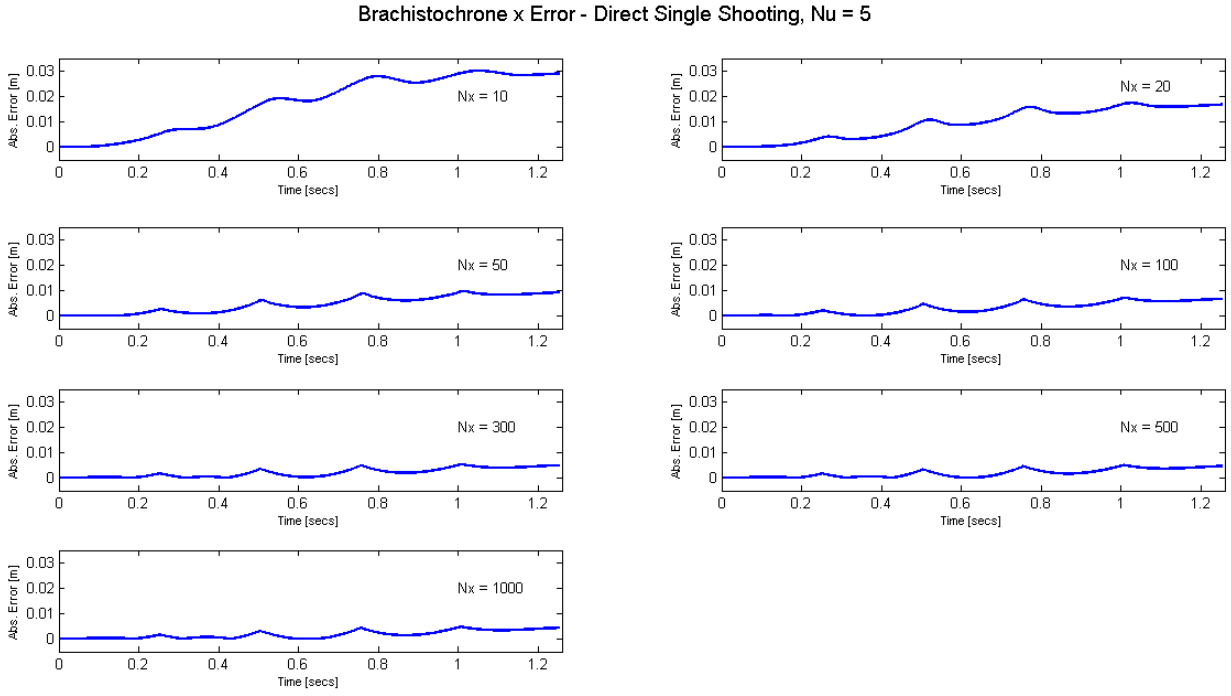

Figure 3.11: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 5$.

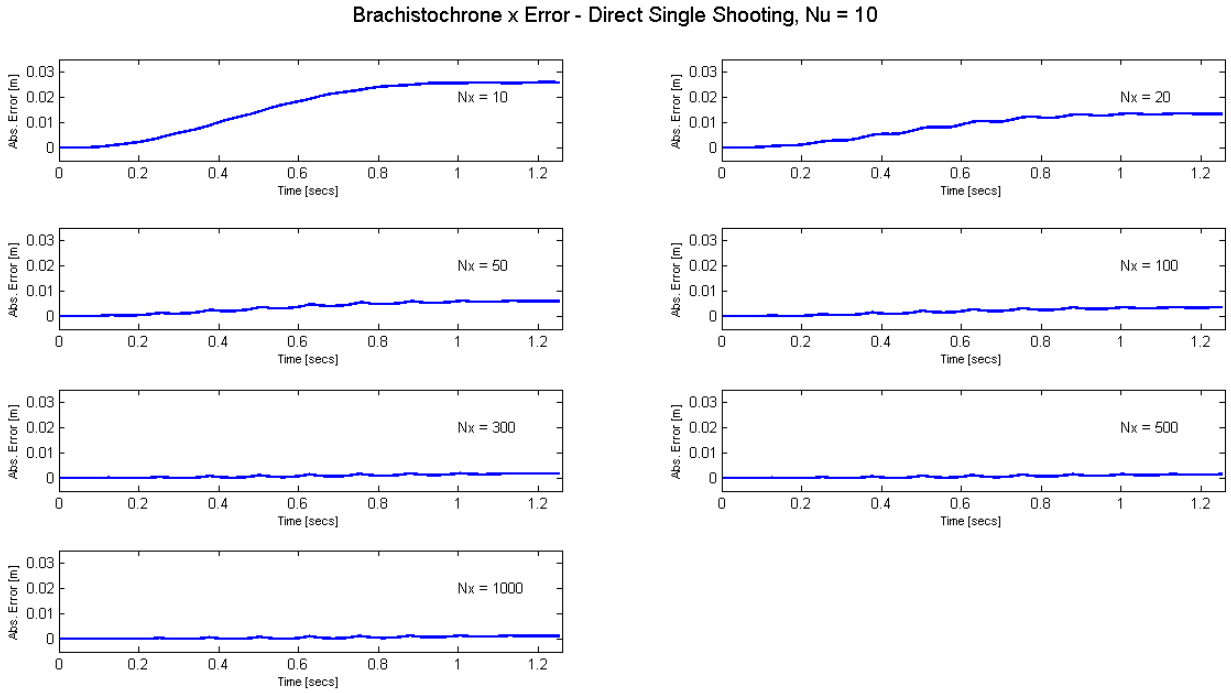

Figure 3.12: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 10$.

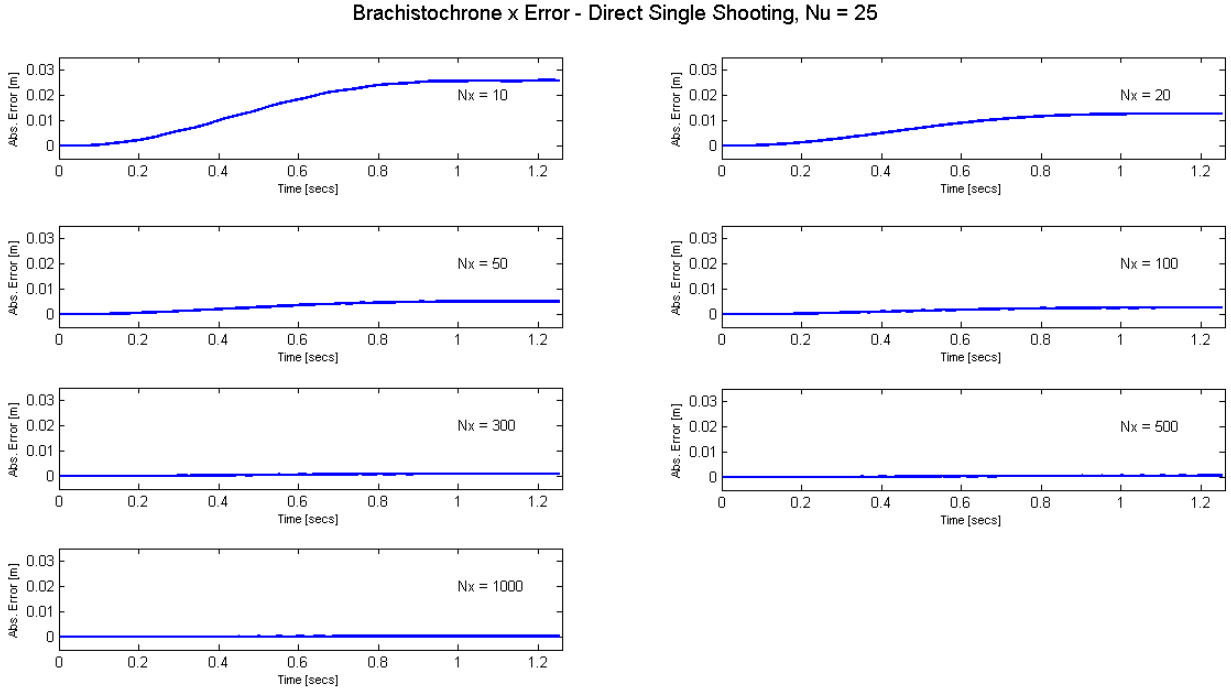

Figure 3.13: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 25$.

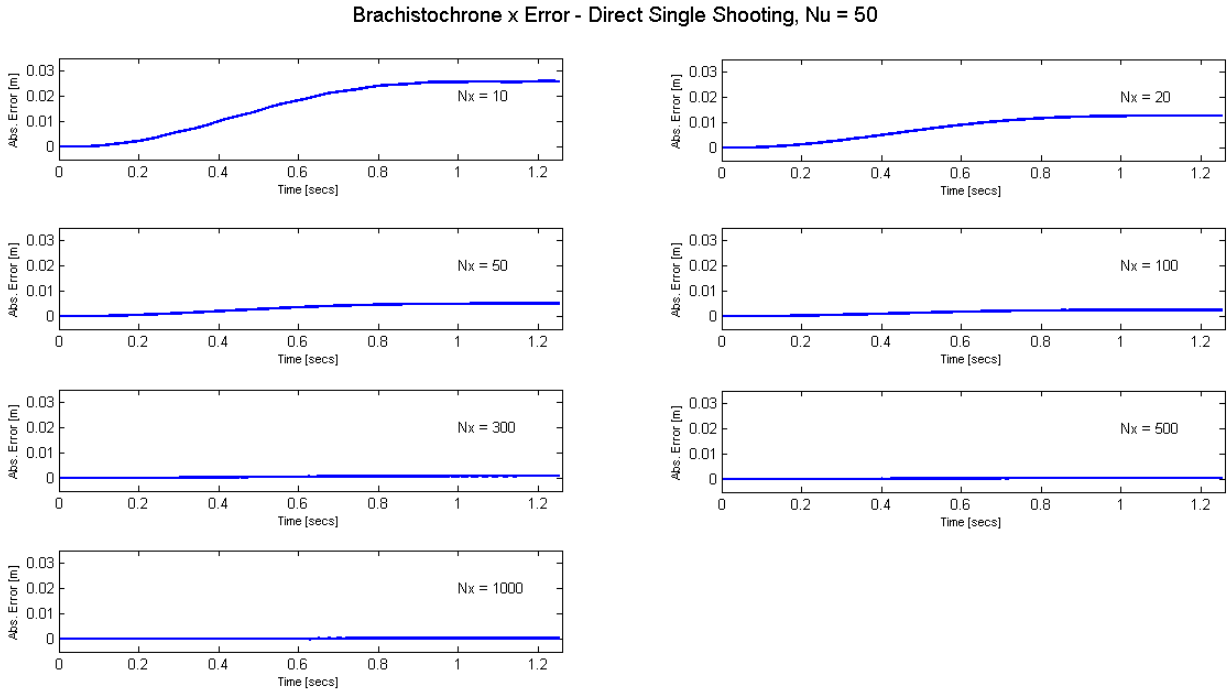

Figure 3.14: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 50$.

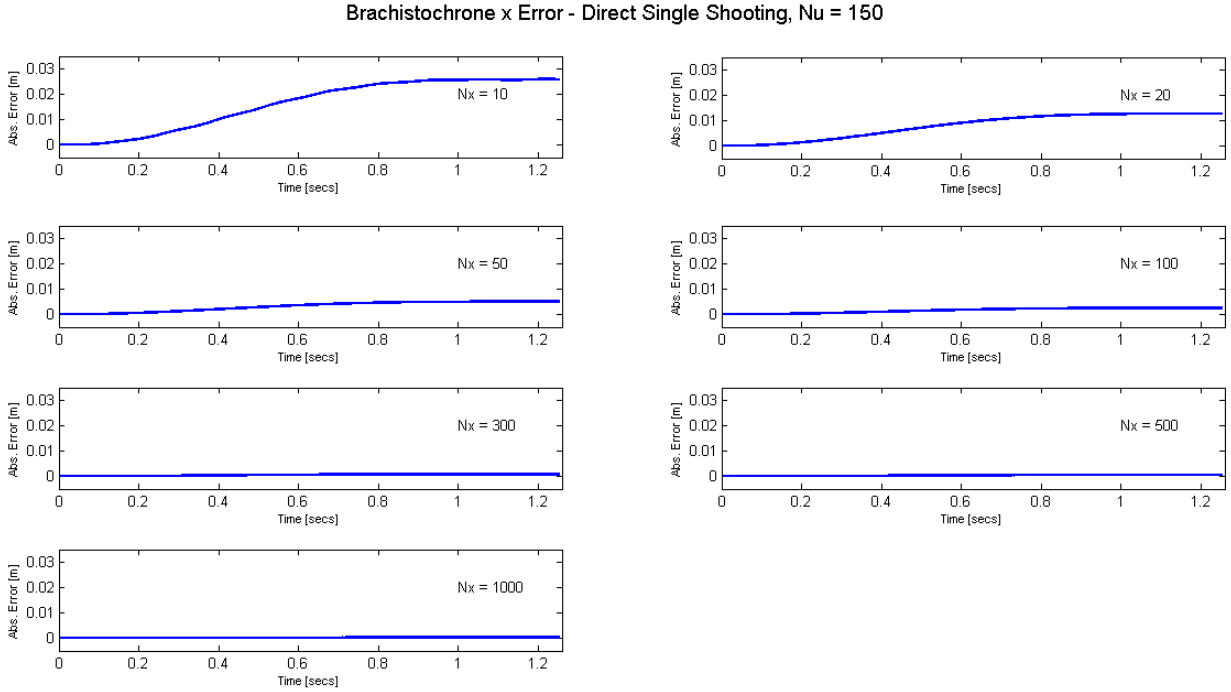

Figure 3.15: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 150$.

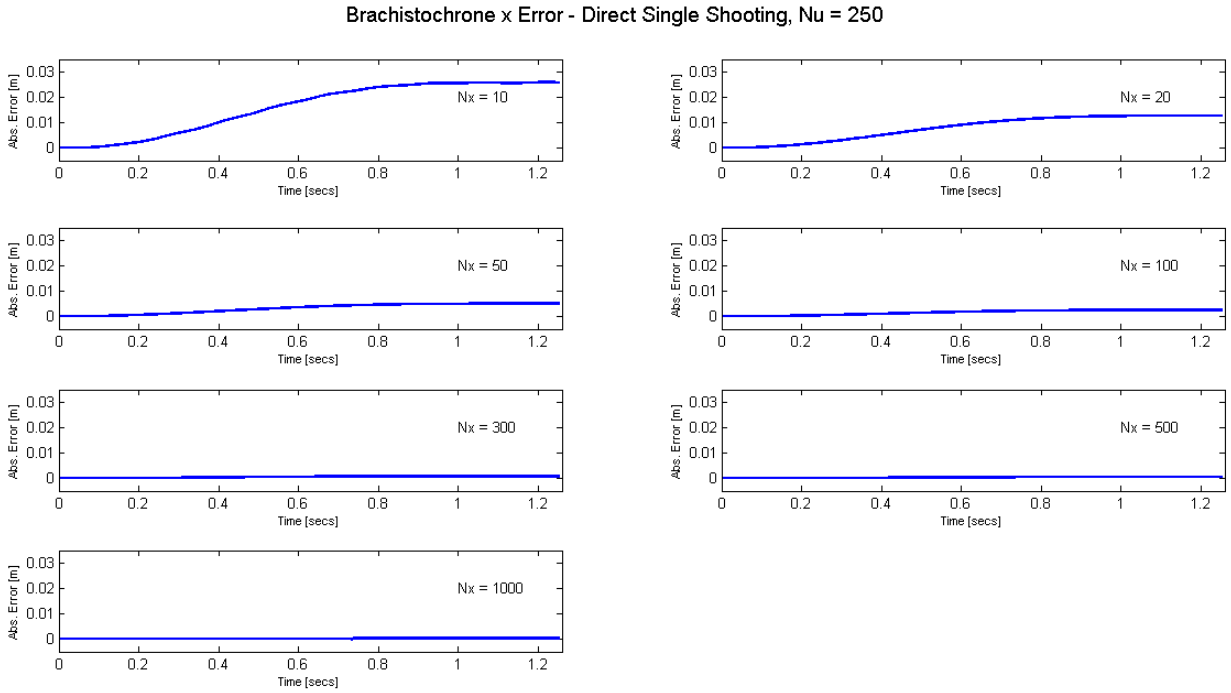

Figure 3.16: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 250$.

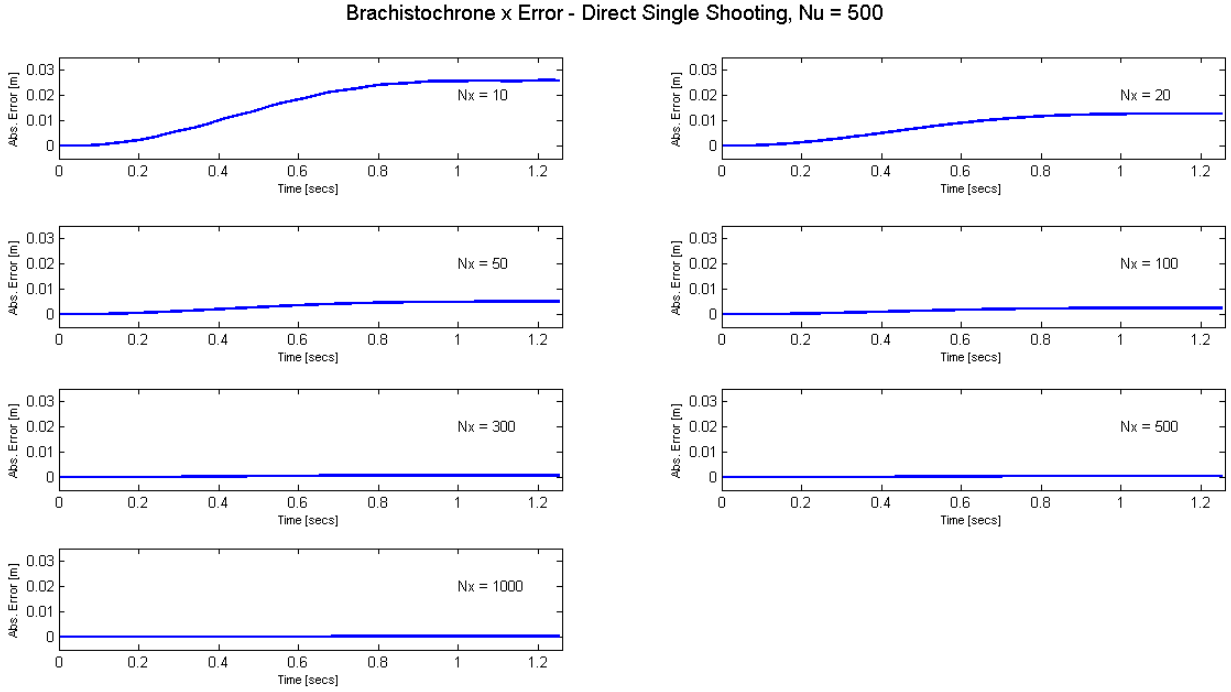

Figure 3.17: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, x-Direction, $N_u = 500$.

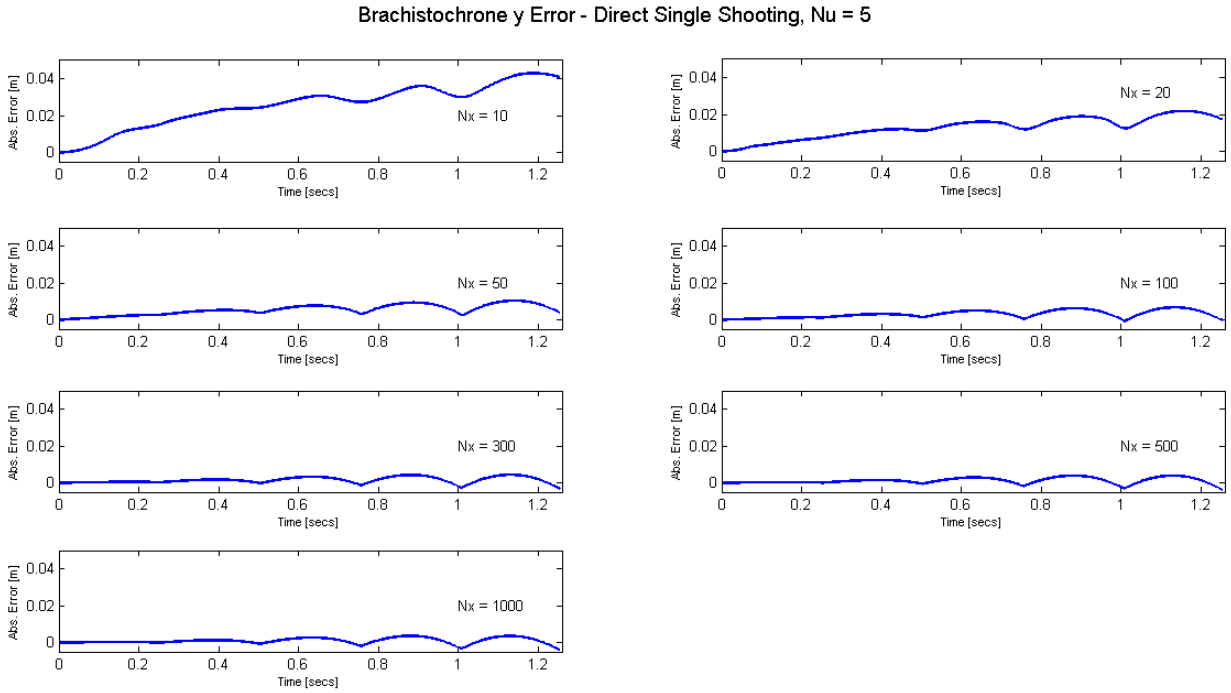

Figure 3.18: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 5$.

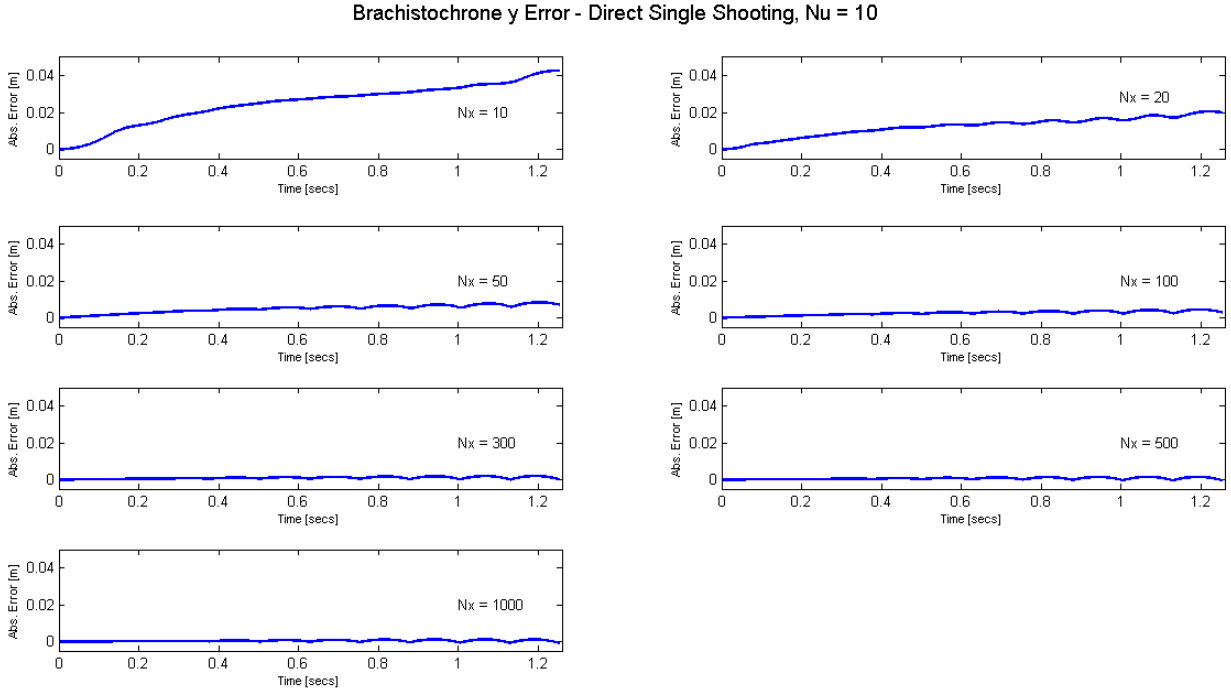

Figure 3.19: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 10$.

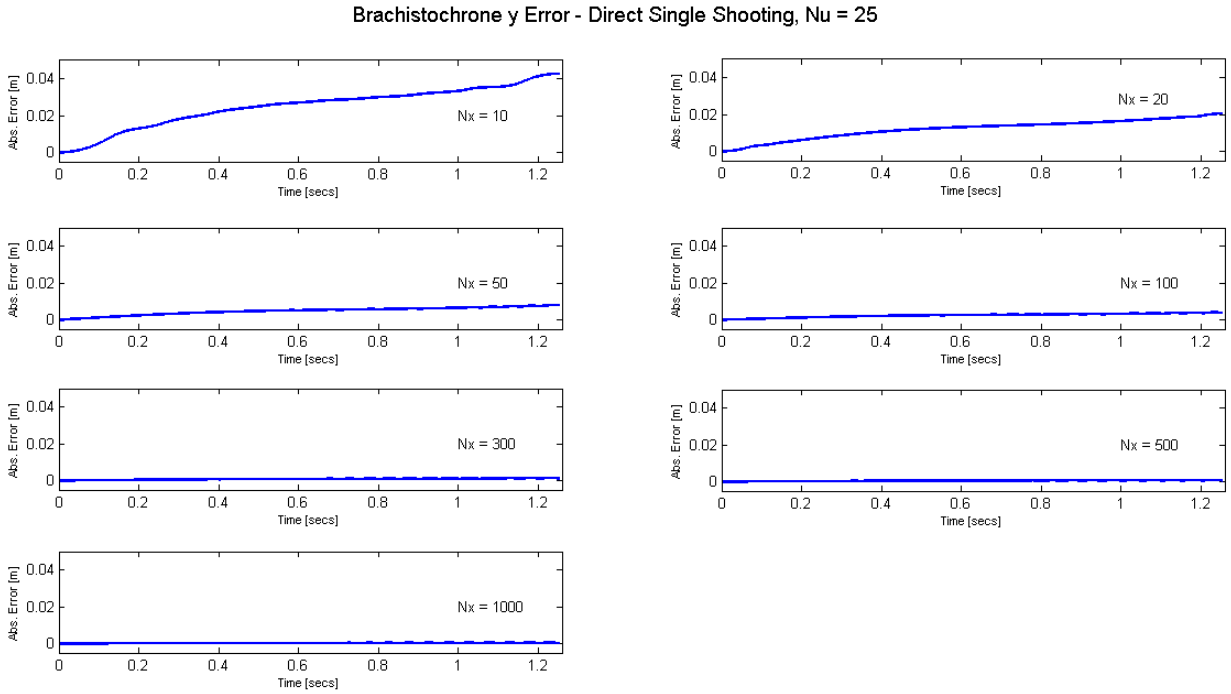

Figure 3.20: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 25$.

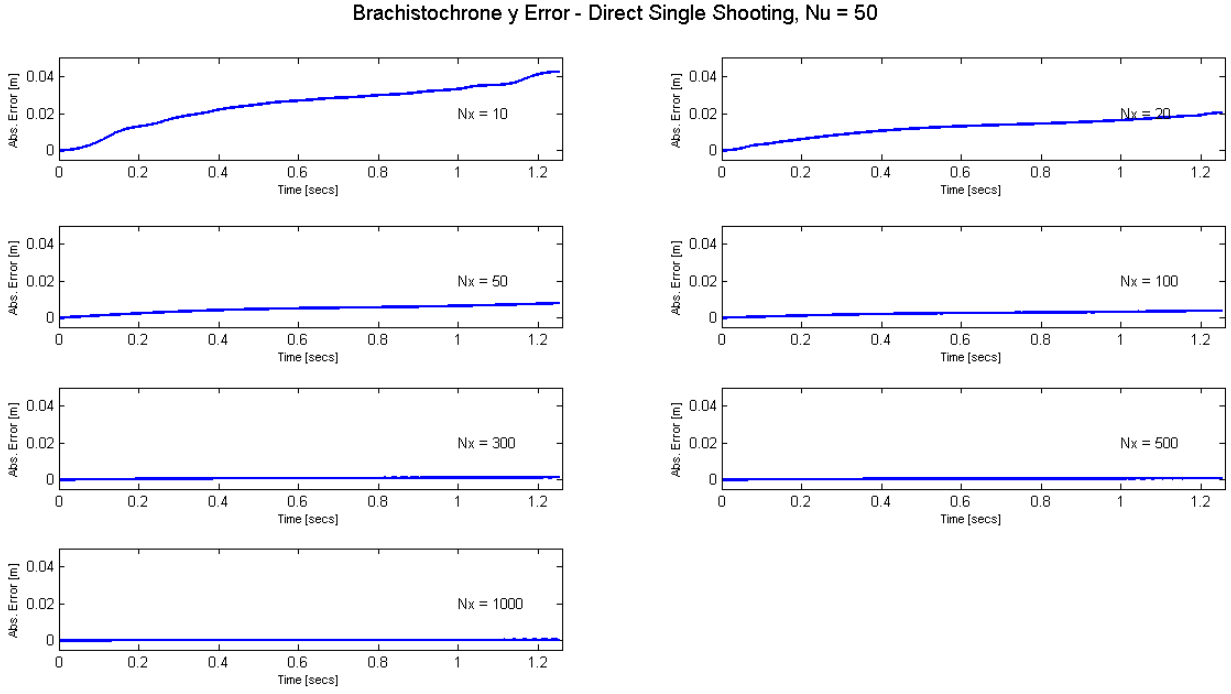

Figure 3.21: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 50$.

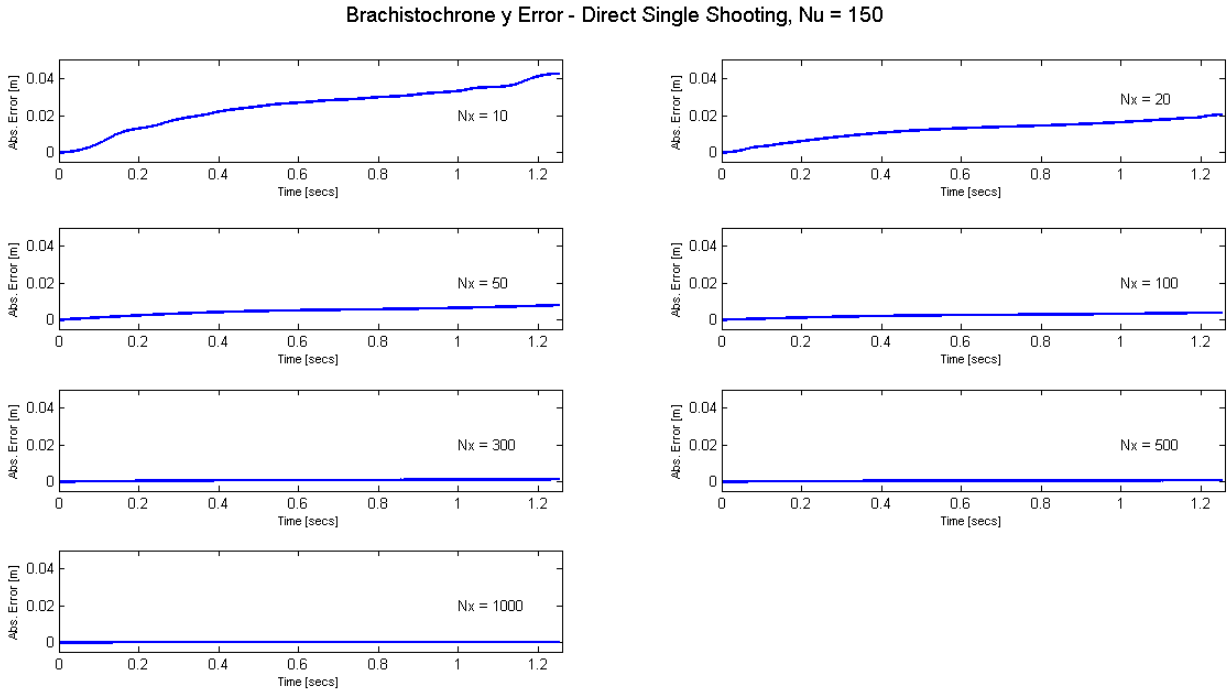

Figure 3.22: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 150$.

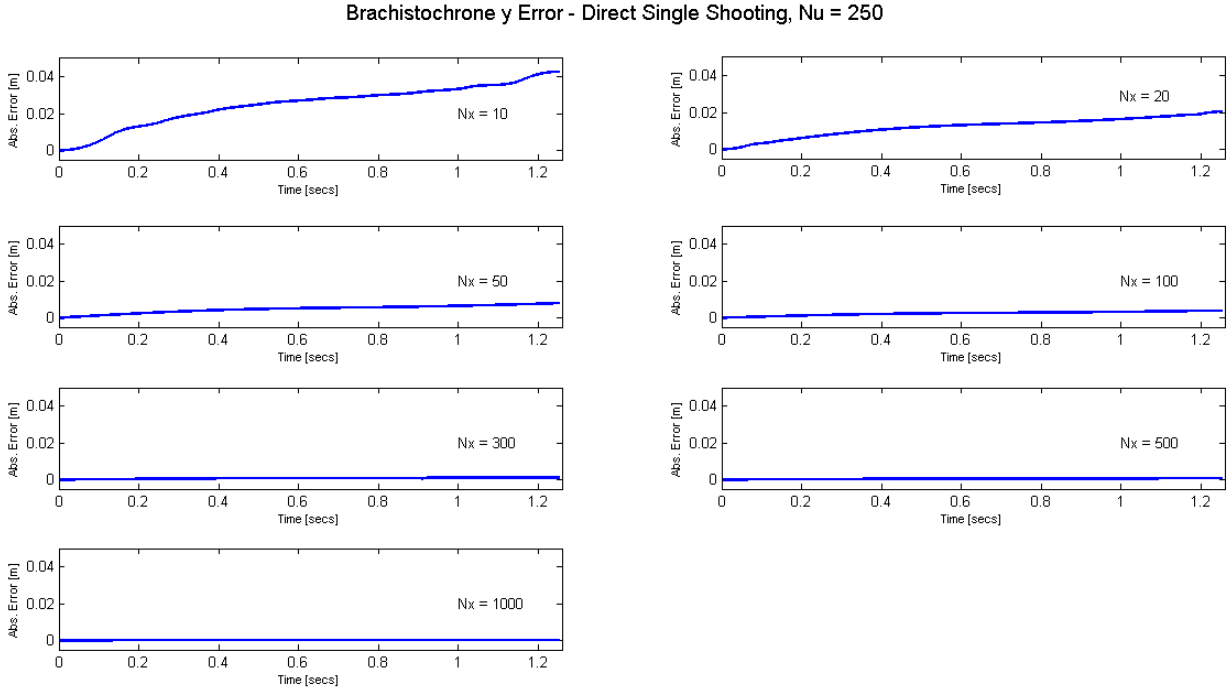

Figure 3.23: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 250$.

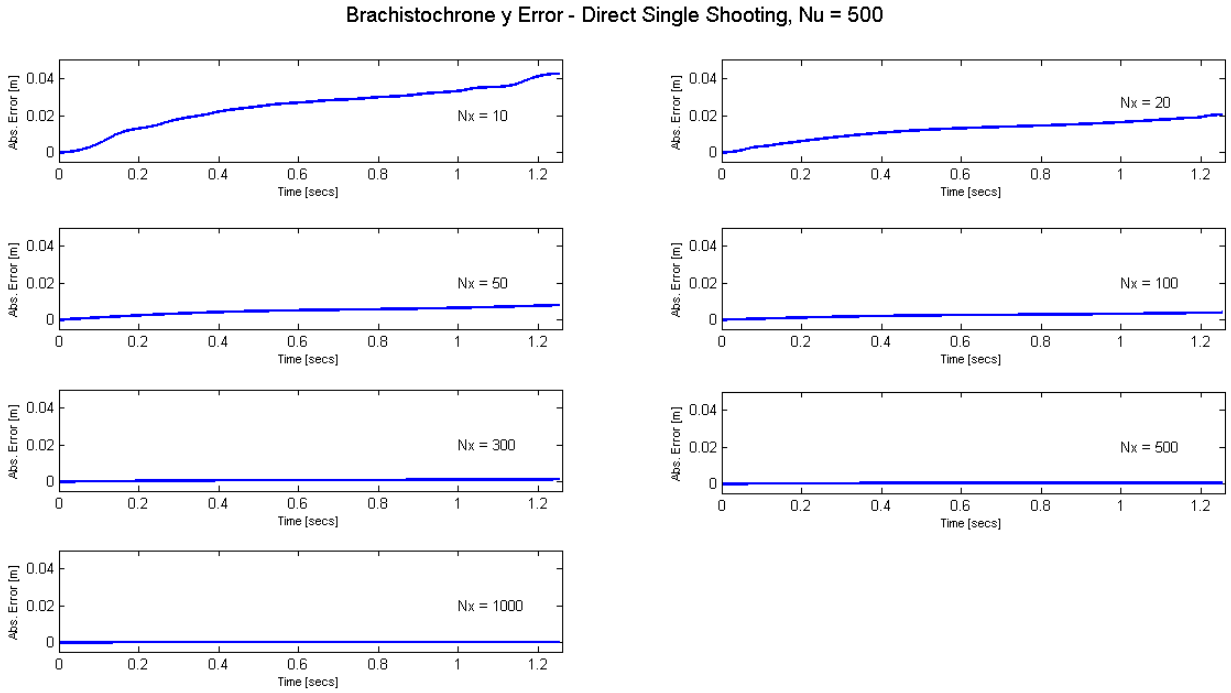

Figure 3.24: Brachistochrone: Direct Single Shooting Absolute Error in Displacement, y-Direction, $N_u = 500$.

3.4.4.1.2 Direct Multiple Shooting Results

The trajectory plots produced by the multiple shooting method are presented in figures 3.25, 3.26, 3.27, 3.28 and 3.29. The results show that increasing the number of sections increases the accuracy of the numerical solution even when the number of control points is as low as two per section. This is further indicated by the magnitude of the errors between the analytical and numerical solutions in the displacement of x given in figures 3.30, 3.31, 3.32, 3.33 and 3.34. The plots show that the errors significantly decrease with increasing M for a given value of N_u. Furthermore given a value of M increasing N_u increases the accuracy of the numerical solution. These trends are also visible in the magnitude of the y displacement errors between numerical and analytical solutions given in figures 3.35, 3.36, 3.37, 3.38 and 3.39. The results indicate that given the same number of discretisation points, the multiple shooting method outperforms the single shooting method in terms of accuracy. In addition all the results pertaining to the multiple shooting method show that there is a point of diminishing returns, as increasing the number of sections beyond 10 can be seen to have very little impact on the accuracy of the solution.

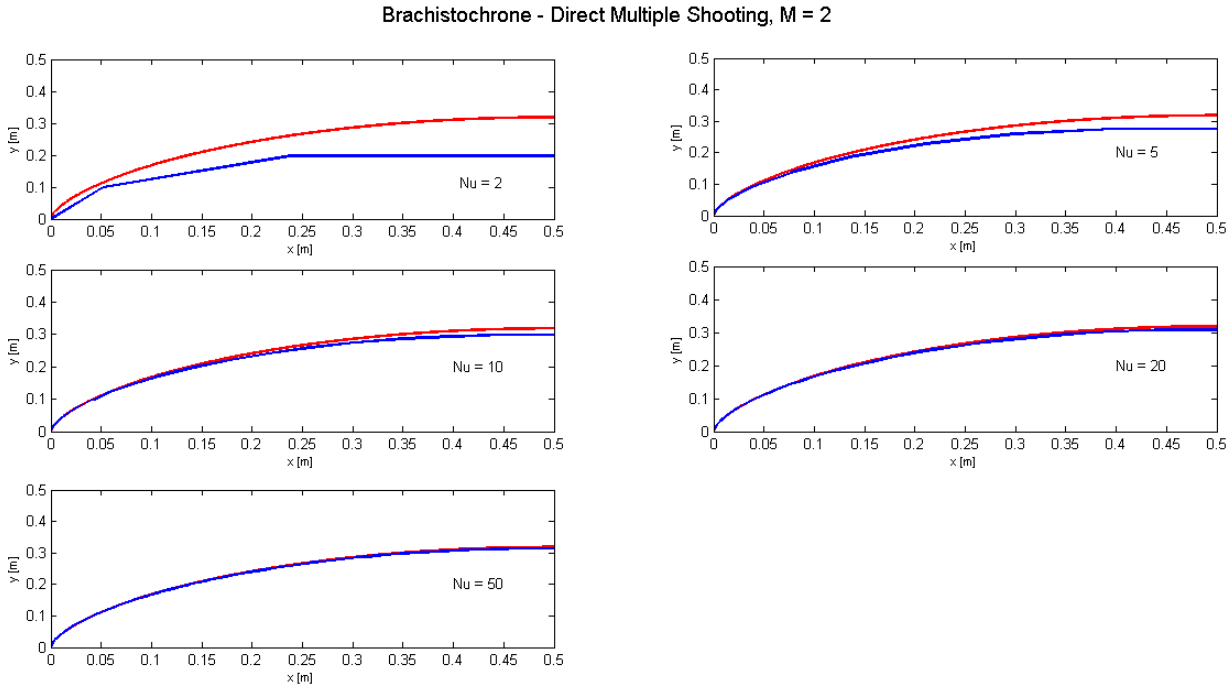

Figure 3.25: Brachistochrone: Direct Multiple Shooting trajectory plots (x vs y), $M = 2$: Analytical Solution (Red) and Numerical Solution (Blue).

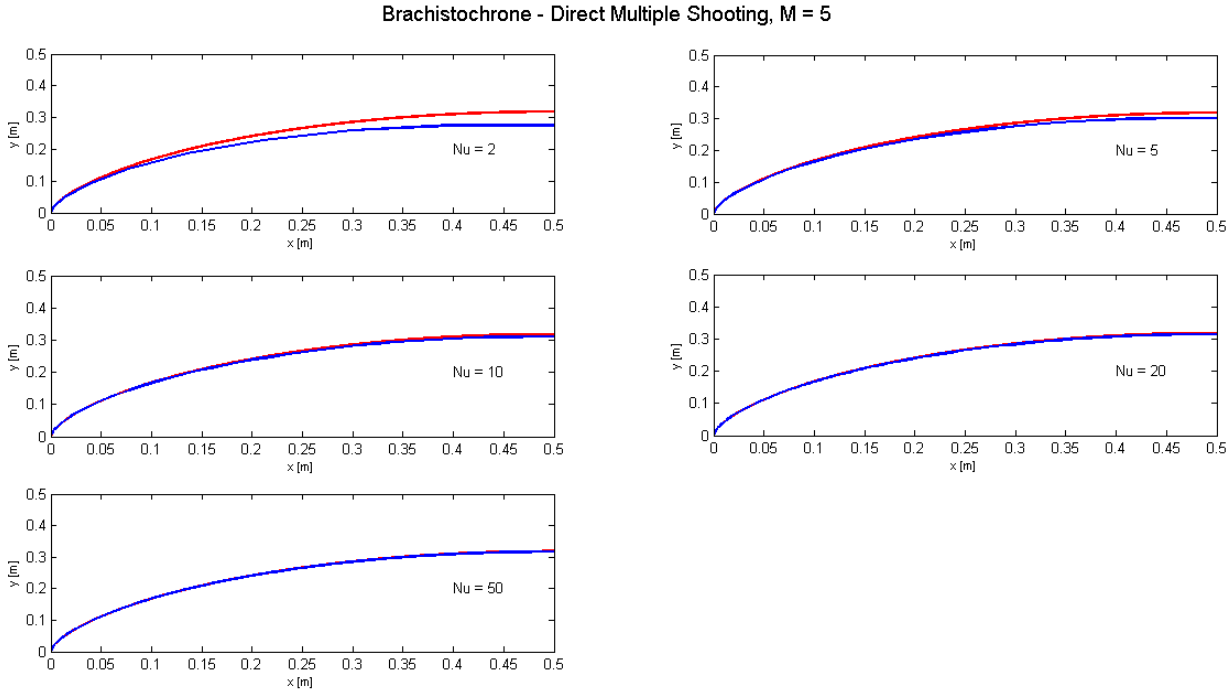

Figure 3.26: Brachistochrone: Direct Multiple Shooting trajectory plots (x vs y), $M = 5$: Analytical Solution (Red) and Numerical Solution (Blue).

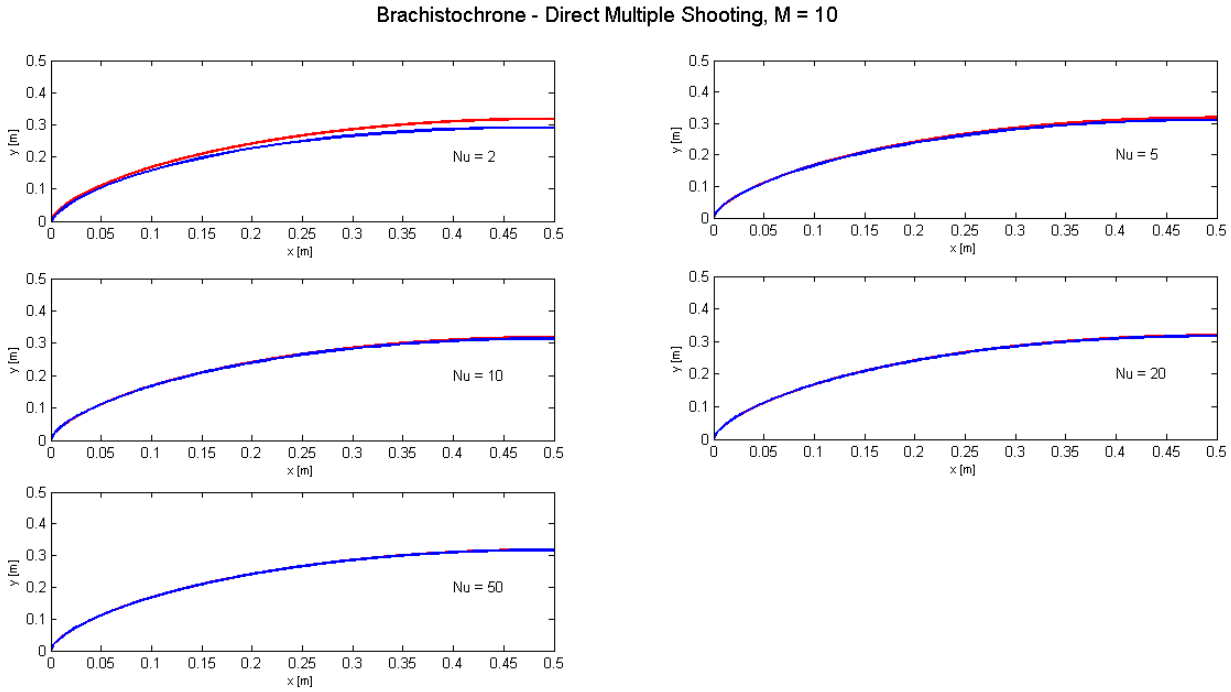

Figure 3.27: Brachistochrone: Direct Multiple Shooting trajectory plots (x vs y), $M = 10$: Analytical Solution (Red) and Numerical Solution (Blue).

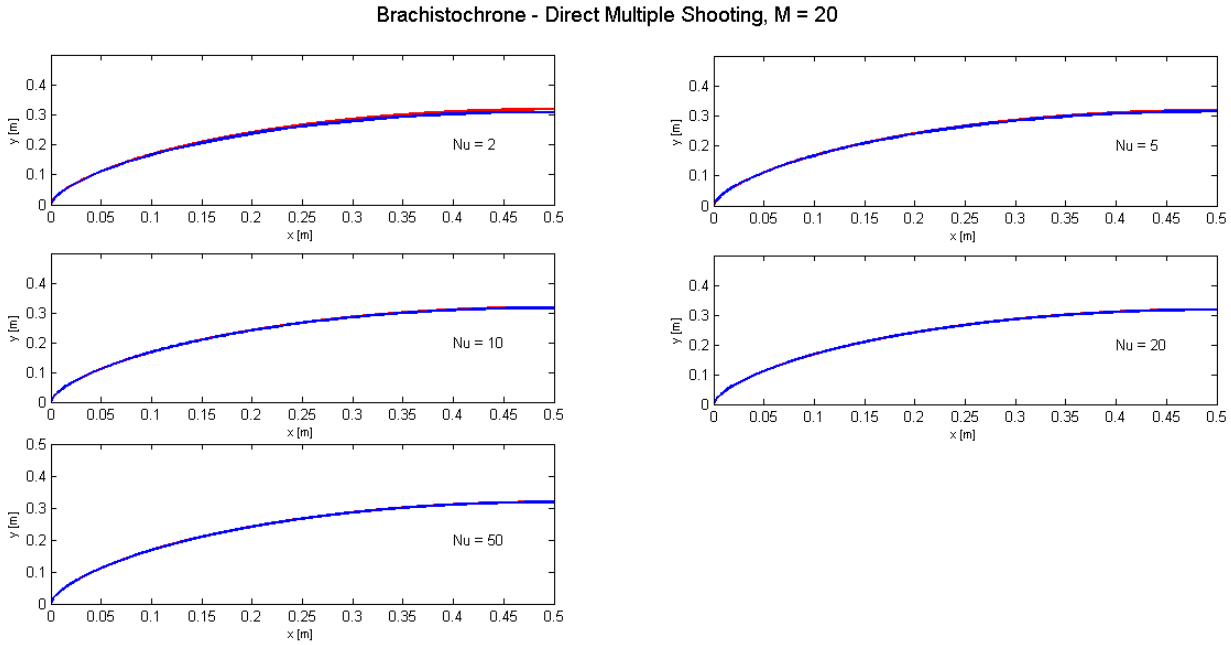

Figure 3.28: Brachistochrone: Direct Multiple Shooting trajectory plots (x vs y), $M = 20$: Analytical Solution (Red) and Numerical Solution (Blue).

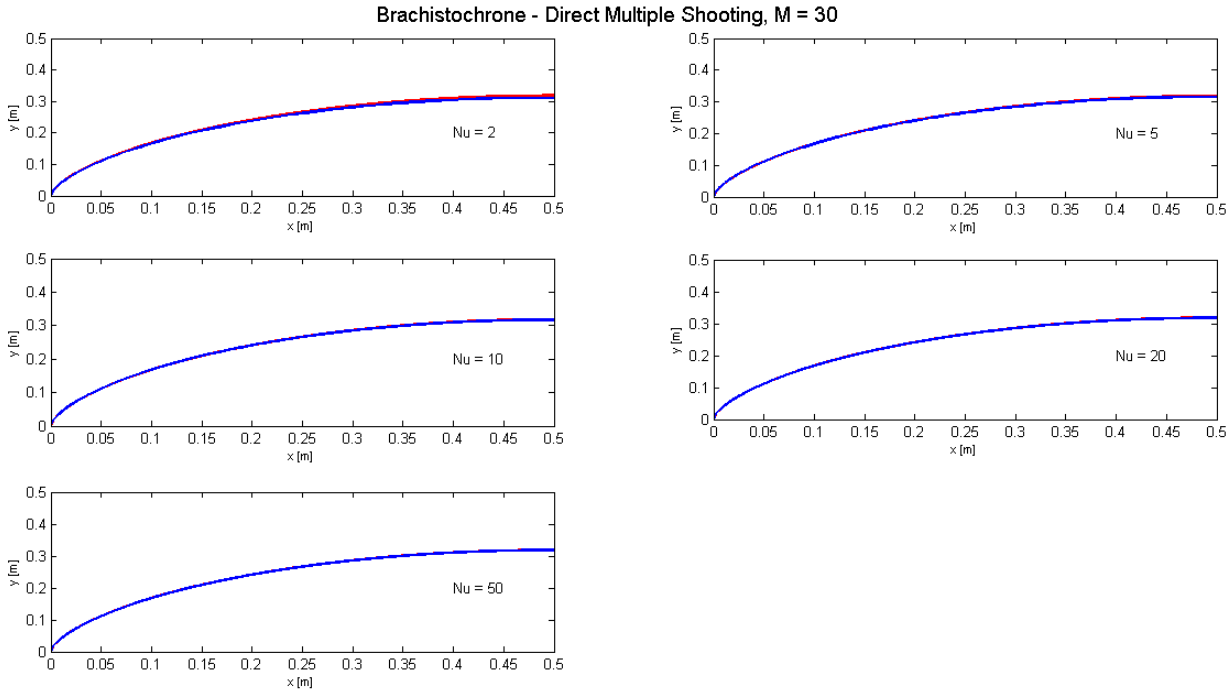

Figure 3.29: Brachistochrone: Direct Single Shooting trajectory plots (x vs y), $M = 30$: Analytical Solution (Red) and Numerical Solution (Blue).

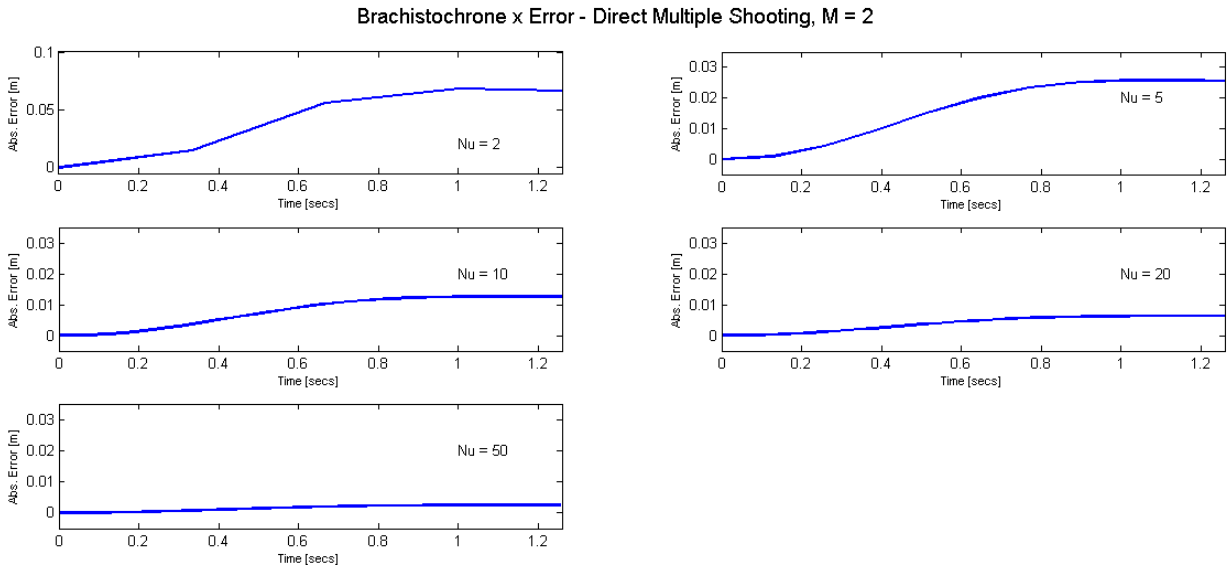

Figure 3.30: Brachistochrone: Direct Multiple Shooting Absolute Error in x-Direction, $M = 2$.

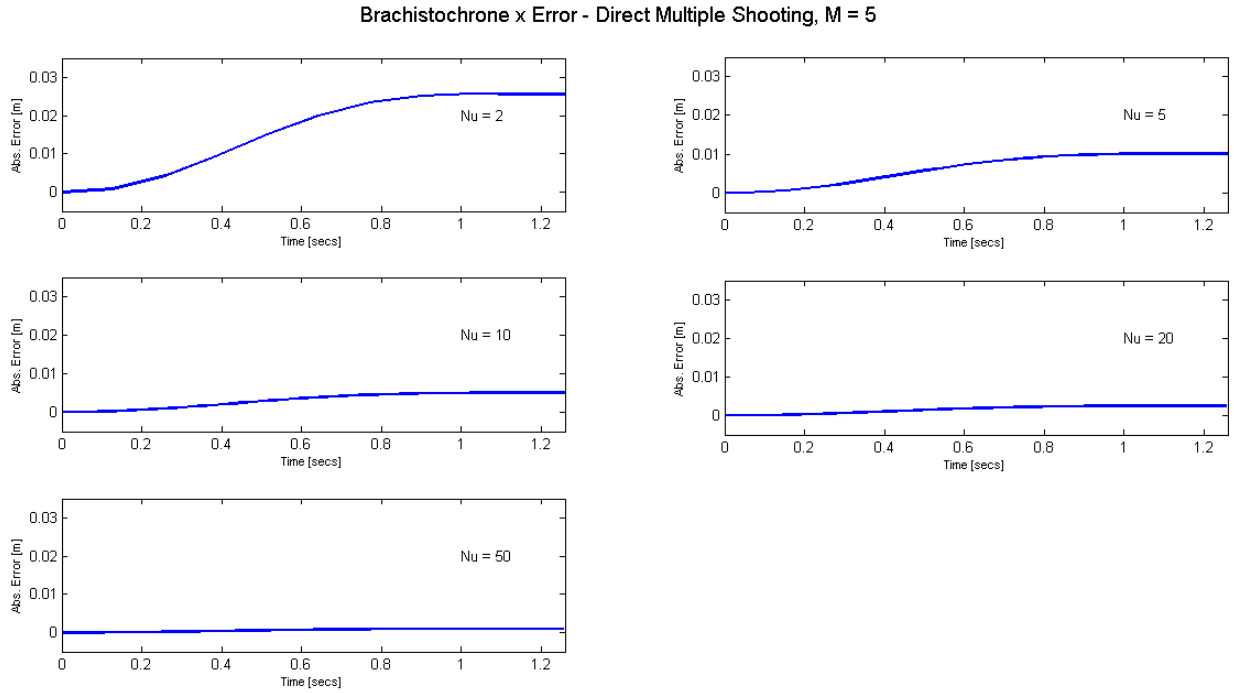

Figure 3.31: Brachistochrone: Direct Multiple Shooting Absolute Error in x-Direction, $M = 5$.

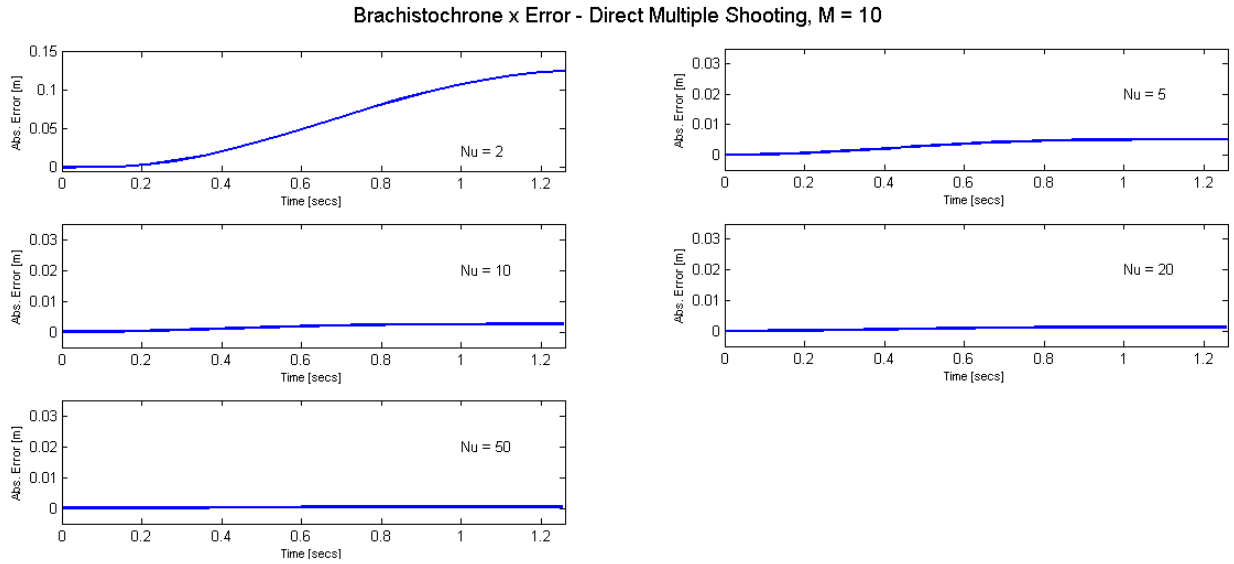

Figure 3.32: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, x-Direction, $M = 10$.

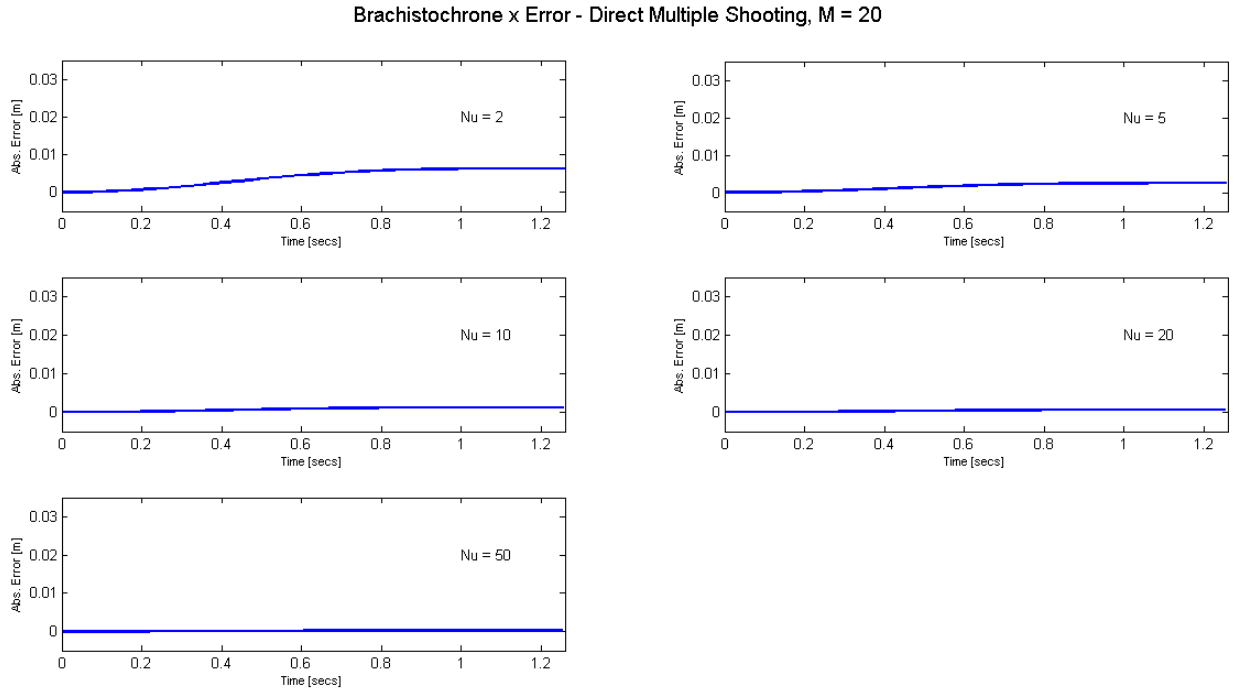

Figure 3.33: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, x-Direction, $M = 20$.

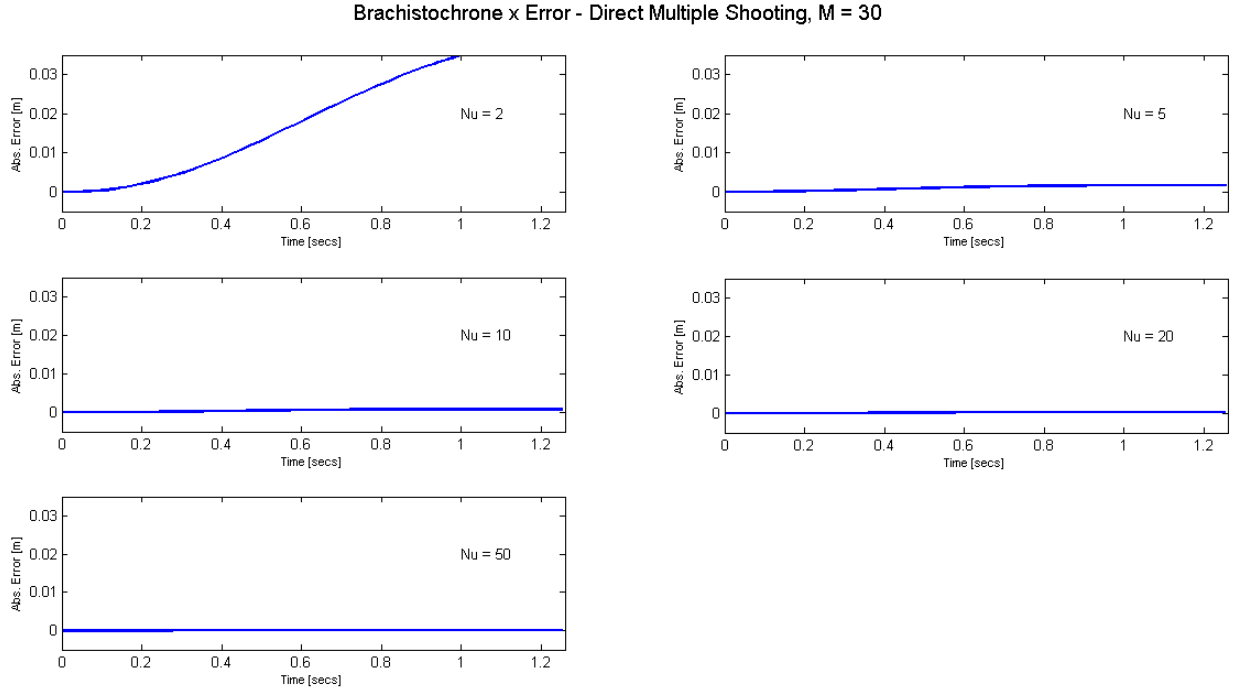

Figure 3.34: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, x-Direction, $M = 30$.

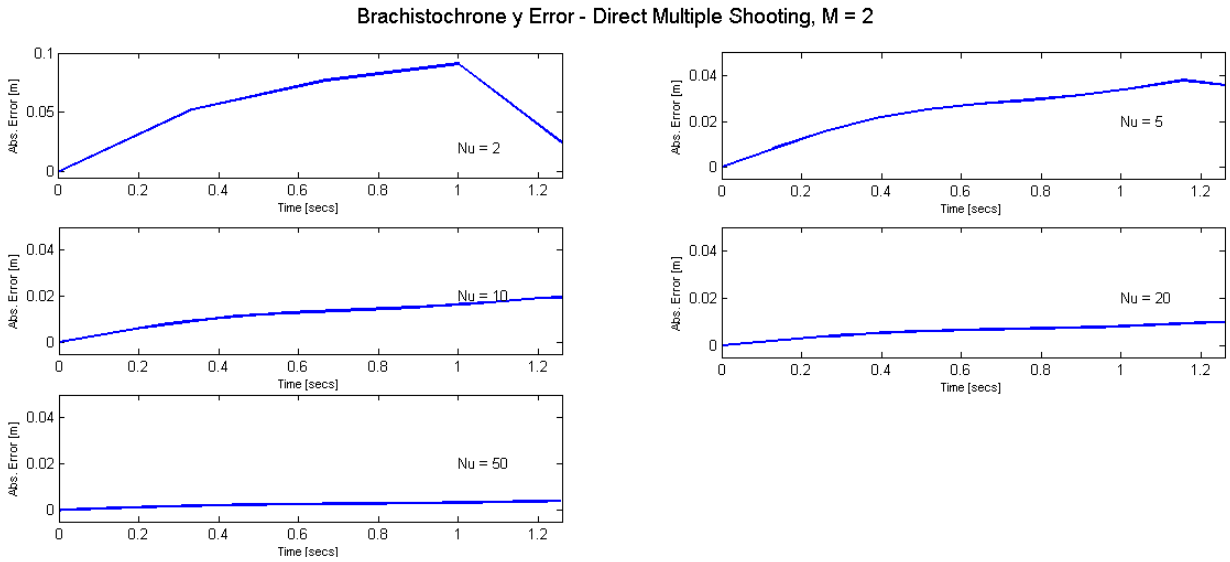

Figure 3.35: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, y-Direction, $M = 2$.

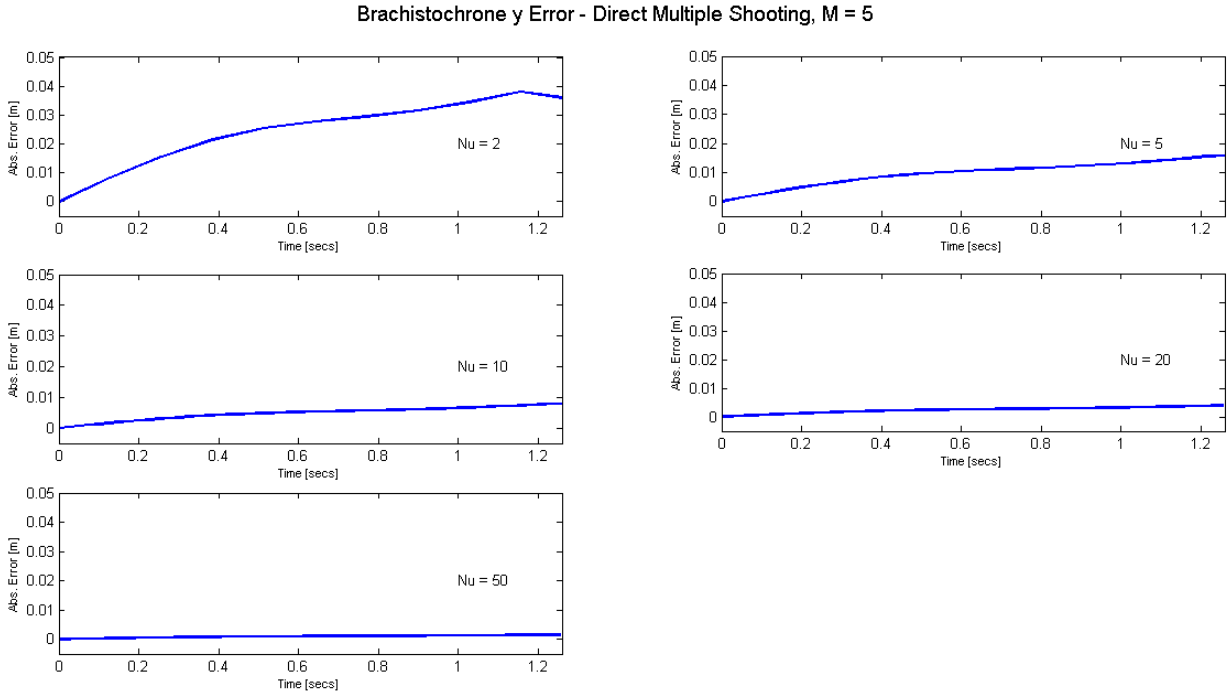

Figure 3.36: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, y-Direction, $M = 5$.

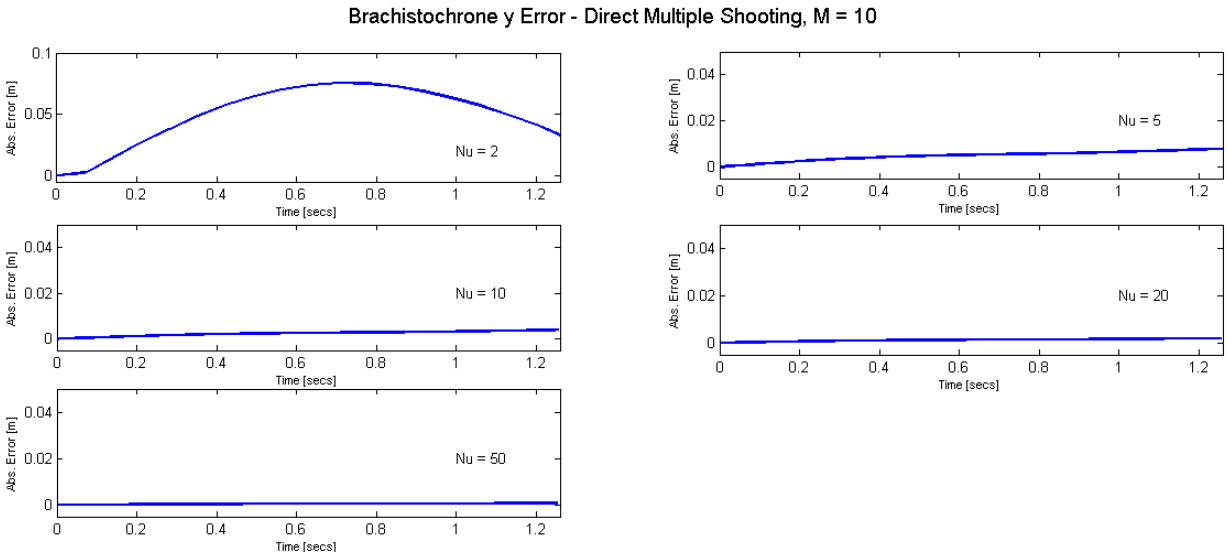

Figure 3.37: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, y-Direction, $M = 10$.

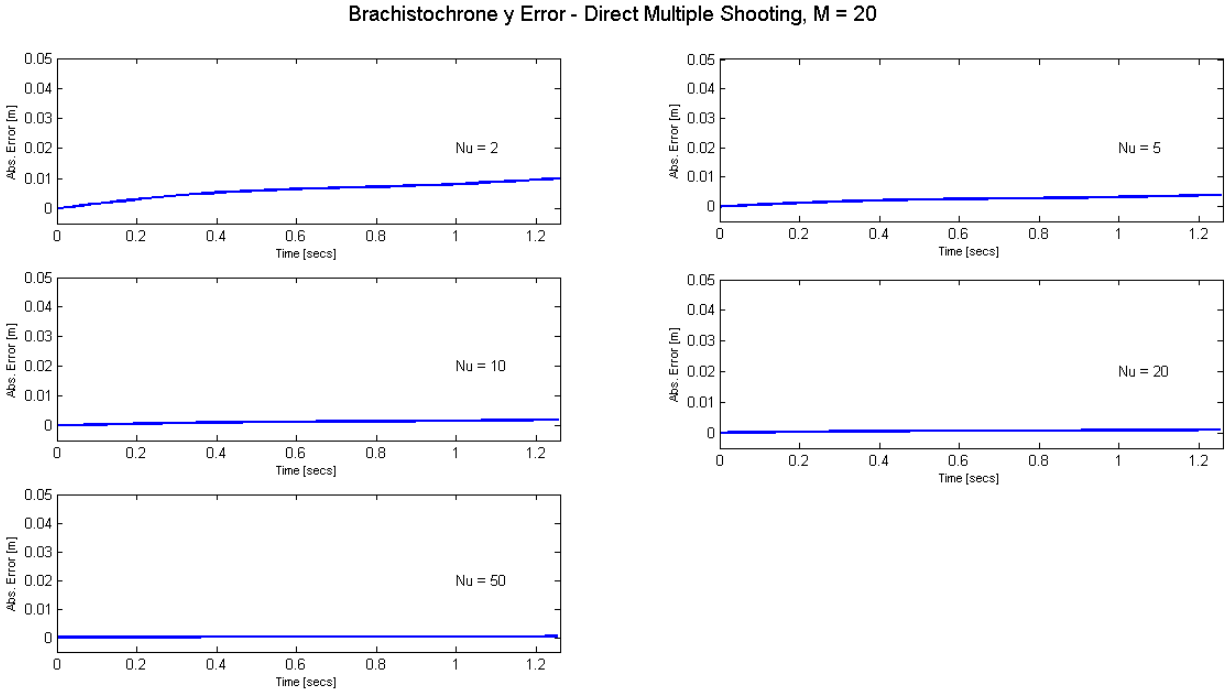

Figure 3.38: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, y-Direction, $M = 20$.

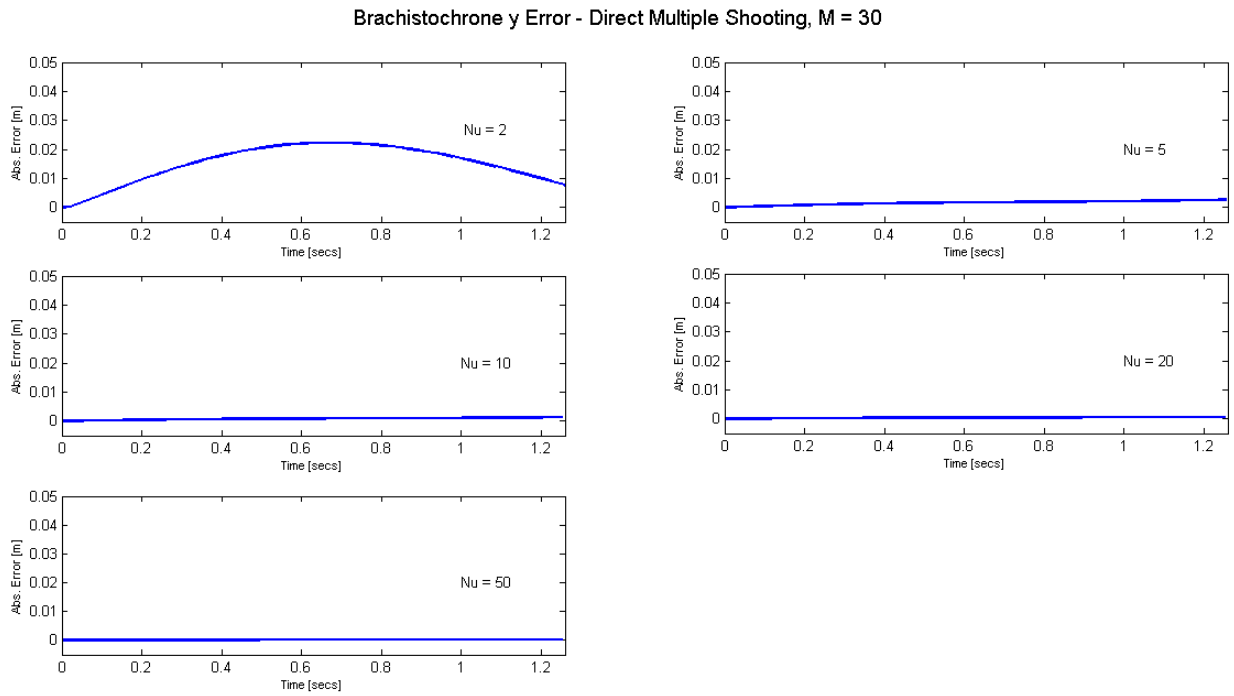

Figure 3.39: Brachistochrone: Direct Multiple Shooting Absolute Error in Displacement, y-Direction, $M = 30$.

3.4.4.1.3 Collocation - Euler Integration Method

Trajectory plots obtained via the collocation Euler integration method are presented in figure 3.40. The results show a high level of compliance between the analytical and the numerical solutions. The magnitude of the errors between the analytical and numerical solutions in the displacement of x and y are provided in figures 3.41 and 3.42 respectively. Results show that the solution converges to the analytical solution when $N = 10$. Increasing the value of N beyond $N = 50$ can be seen to have little or no effect on the accuracy of the results.

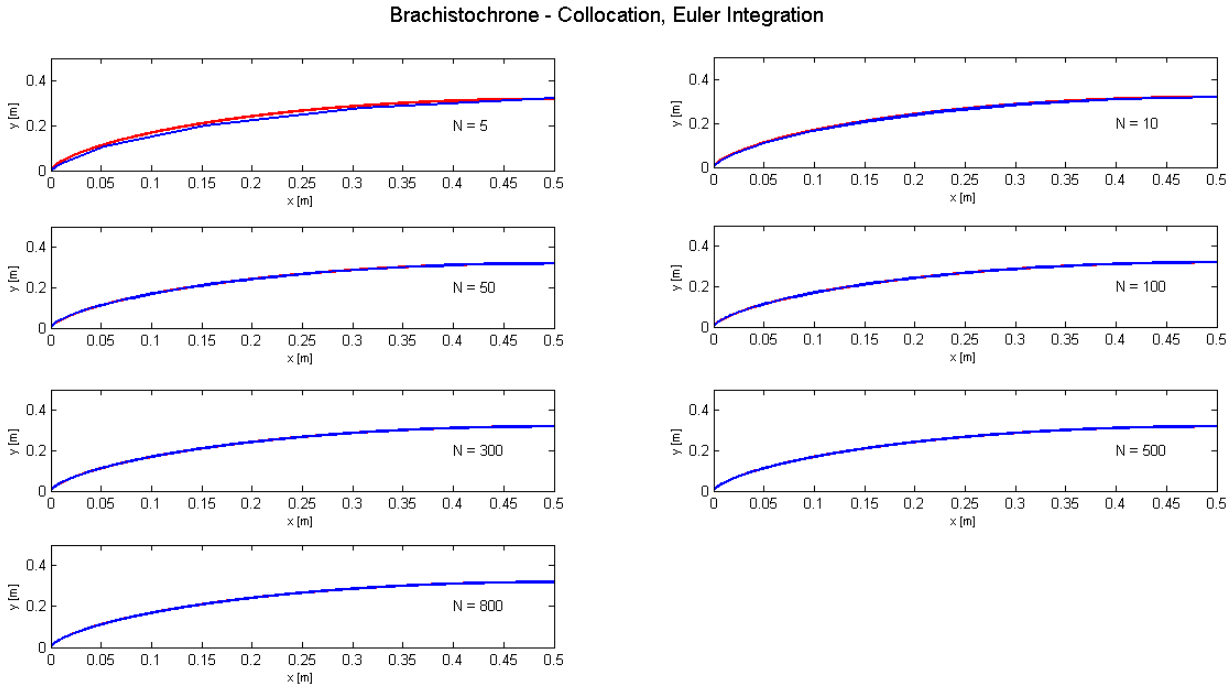

Figure 3.40: Brachistochrone: Collocation Trajectory Plots (x vs y), Analytical Solution (Red) and Numerical Solution (Blue).

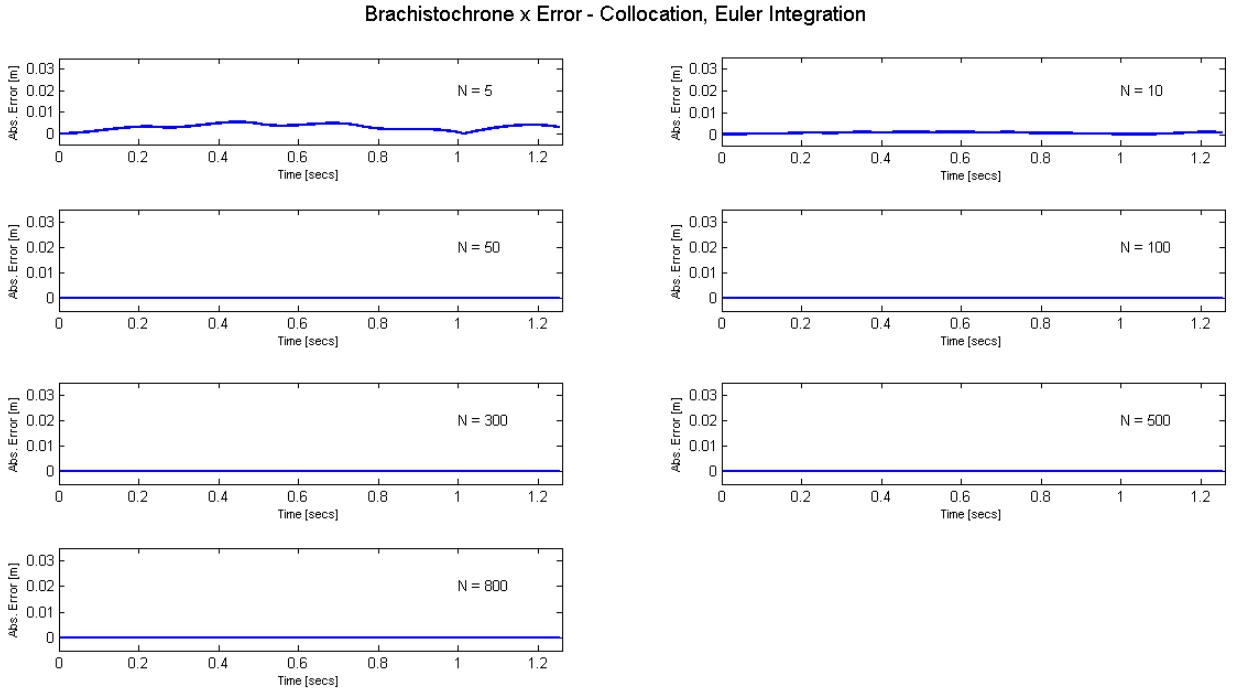

Figure 3.41: Brachistochrone: Collocation Absolute Error in Displacement, x-Direction.

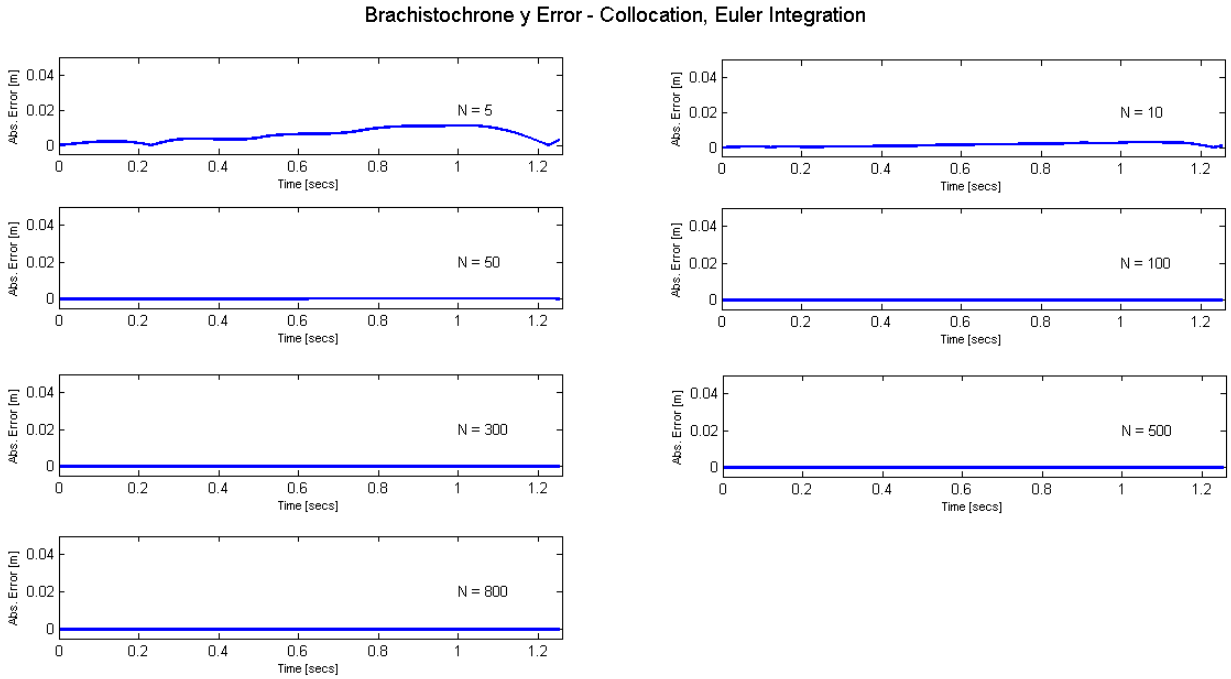

Figure 3.42: Brachistochrone: Collocation Absolute Error in Displacement, y-Direction.

3.4.4.1.4 Pseudospectral Collocation The pseudospectral (derivative based collocation) method produced the most accurate results as the error between the numerical and analytical solutions decreased compared to the collocation Euler integration method for a given value of N. This is evident from the trajectory plots provided in figure 3.43. The magnitude of the errors between the analytical and numerical solutions in the displacements of x and y shown in figures 3.44 and 3.45 respectively help to further support this observation. Again increasing the value of N beyond 50 was seen to have little or no effect on the accuracy of the numerical solution. In fact the pseudospectral solution showed that even when $N = 10$ the numerical solution is in compliance with the analytical solution.

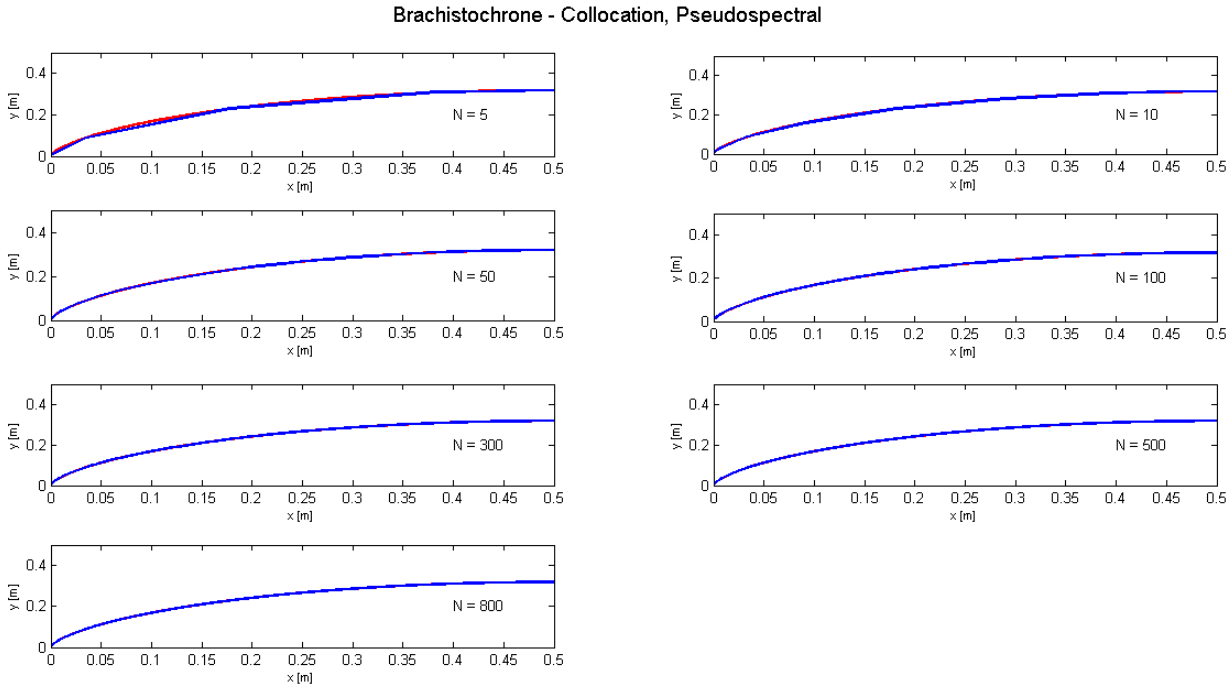

Figure 3.43: Brachistochrone: Pseudospectral Trajectory Plots (x vs y), Analytical Solution (Red) and Numerical Solution (Blue).

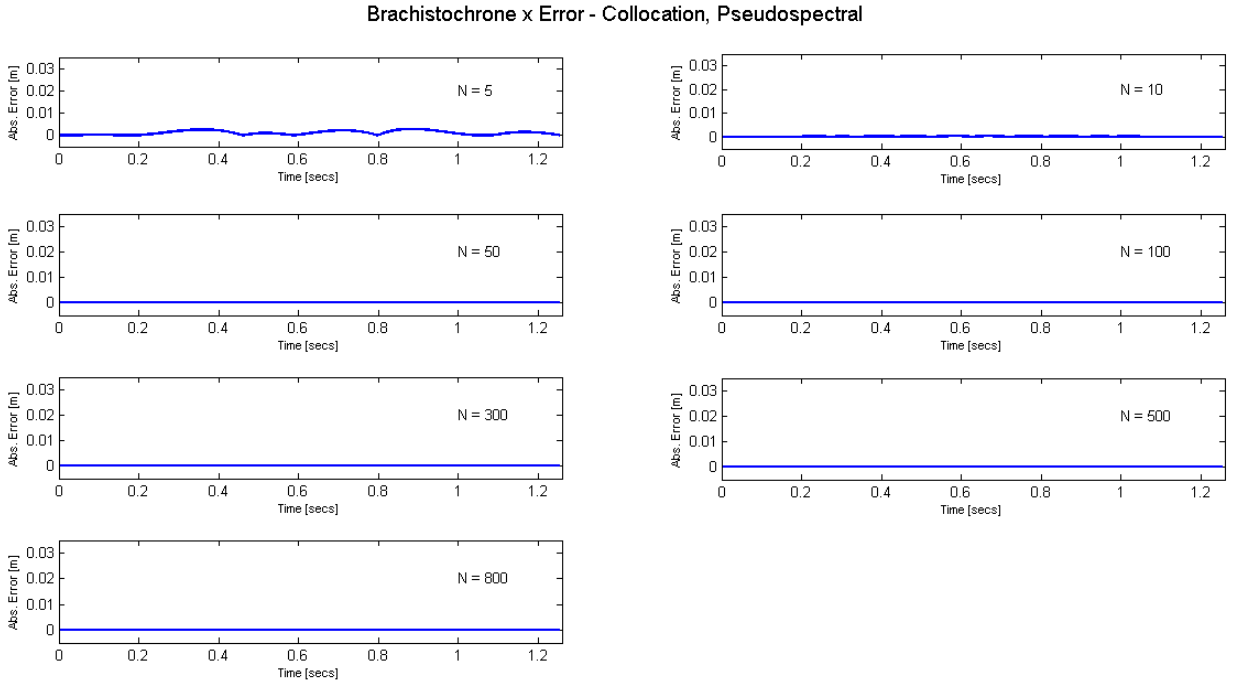

Figure 3.44: Brachistochrone: Pseudospectral Absolute Error in Displacement, x-Direction.

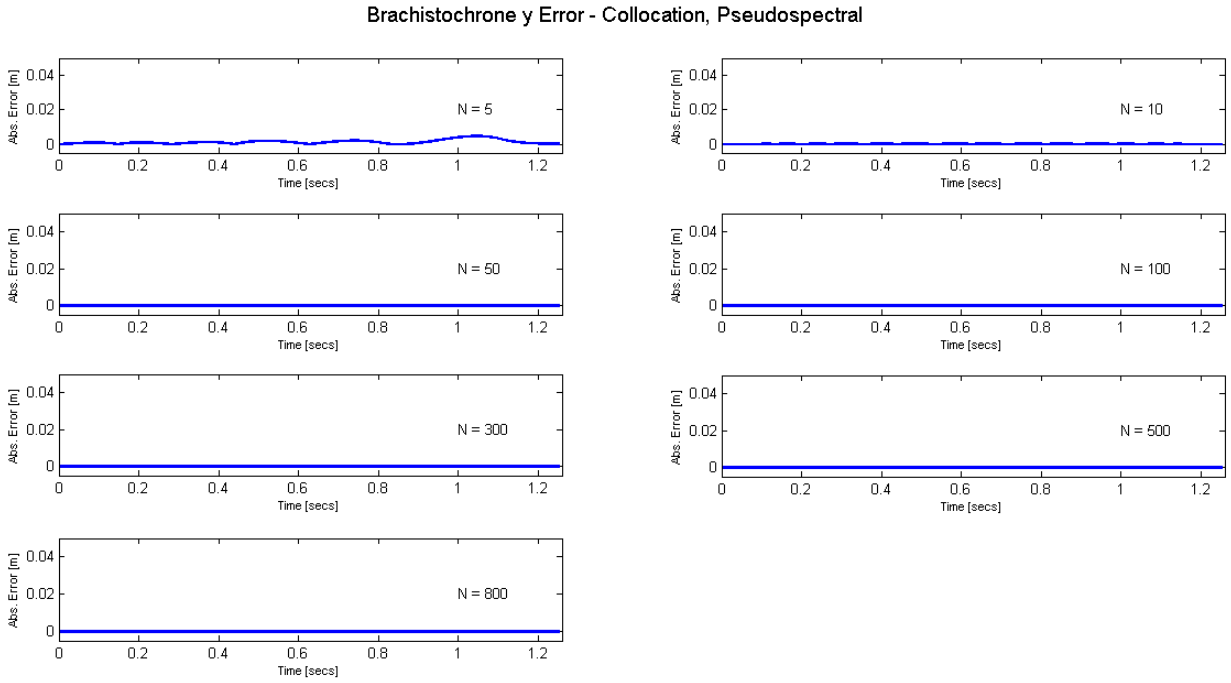

Figure 3.45: Brachistochrone: Pseudospectral Absolute Error in Displacement, y-Direction.

3.4.4.2 Analysis of CPU Time

From the results given above the Pseudospectral method produced the most accurate solution for the least number of discretisation points. For this reason the results produced by the Pseudospectral method with $N = 50$ points is used as the nominal solution for the following analysis.

The time taken to reach $x_f = 0.5$ in the solution of the analytical problem given by equation (3.39) is $t_f = 1.2533\,\text{secs}$. The nominal solution produced the same final time. Figures given in 3.46 through to 3.59 show the central processing unit (CPU) time taken to reach an optimal solution as a percentage of the nominal. The error plots show the percentage error between the optimal solution produced by each method and the nominal solution.

3.4.4.2.1 Direct Single Shooting Results

Figure 3.46 shows the CPU times and t_f errors for the direct single shooting method for $N_u = 5$. The plots show that for $N_u = 5$ the CPU time is less than the nominal for all N_x however the correct final time is unattainable. The same trend was present when N_u was increased to 10. For $N_u = 25$, 1000 state points were required to reach the correct final value however this took twice as long as the nominal to reach the solution. As the values of N_u increased it was found

that the correct final time was only achievable for very large numbers of N_x taking up to 5 times more CPU time compared to the nominal case to reach the correct solution. The CPU percentages are also tabulated in table 3.1.

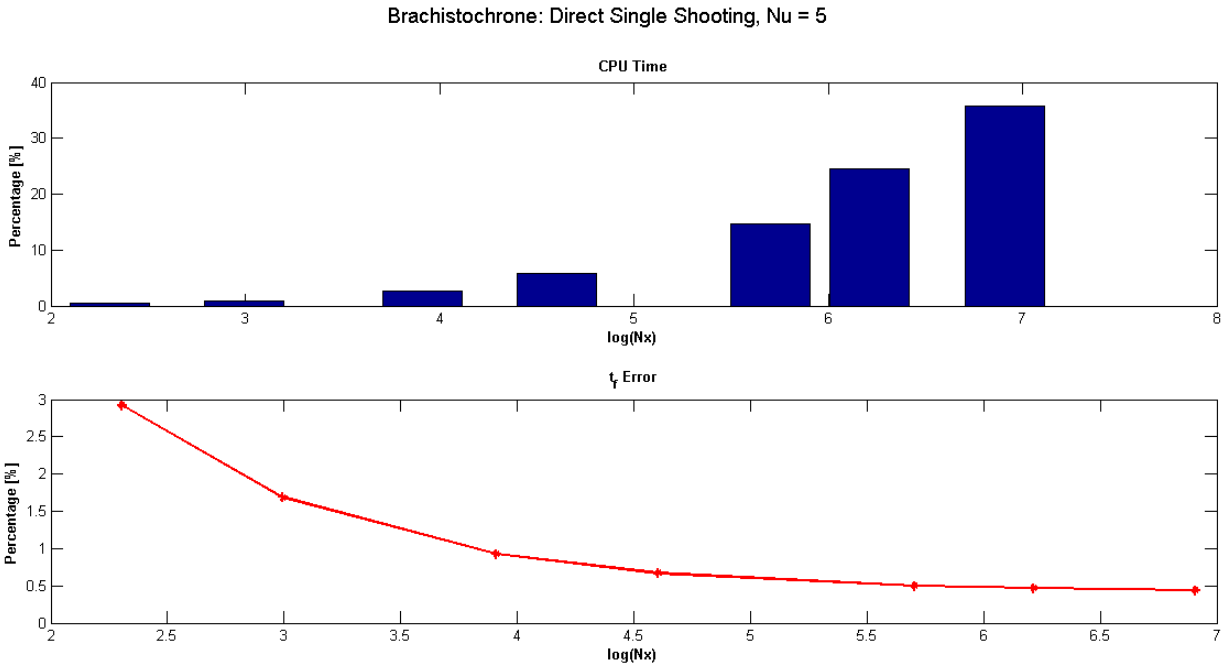

Figure 3.46: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 5$

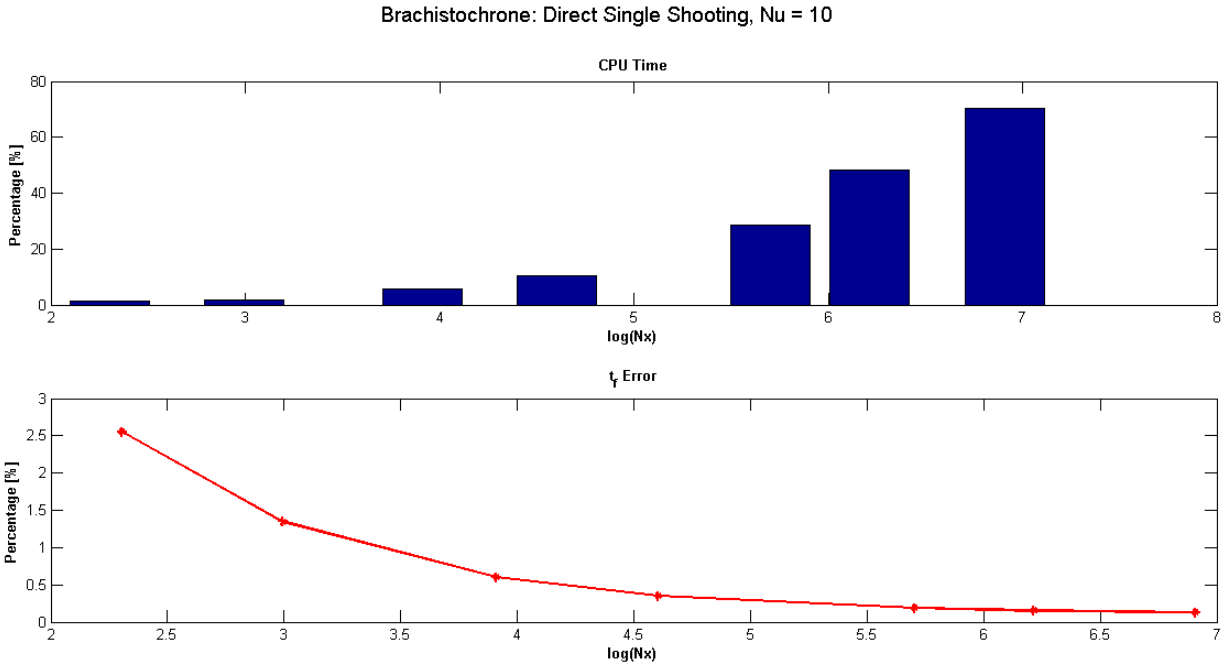

Figure 3.47: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 10$

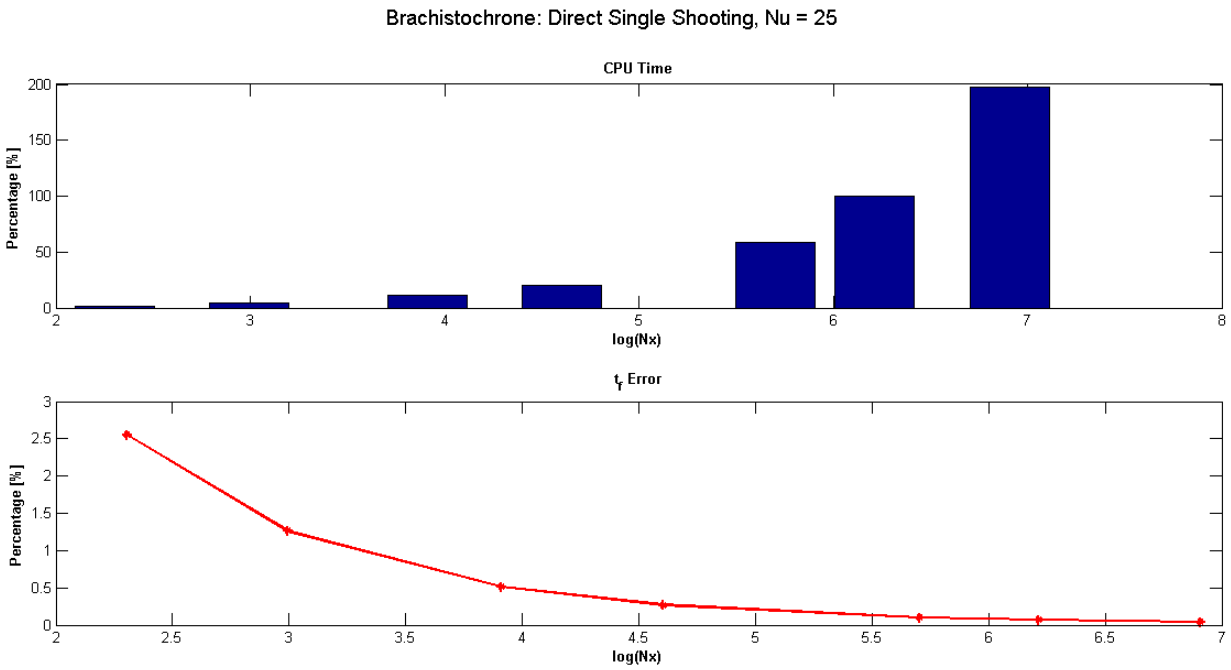

Figure 3.48: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 25$

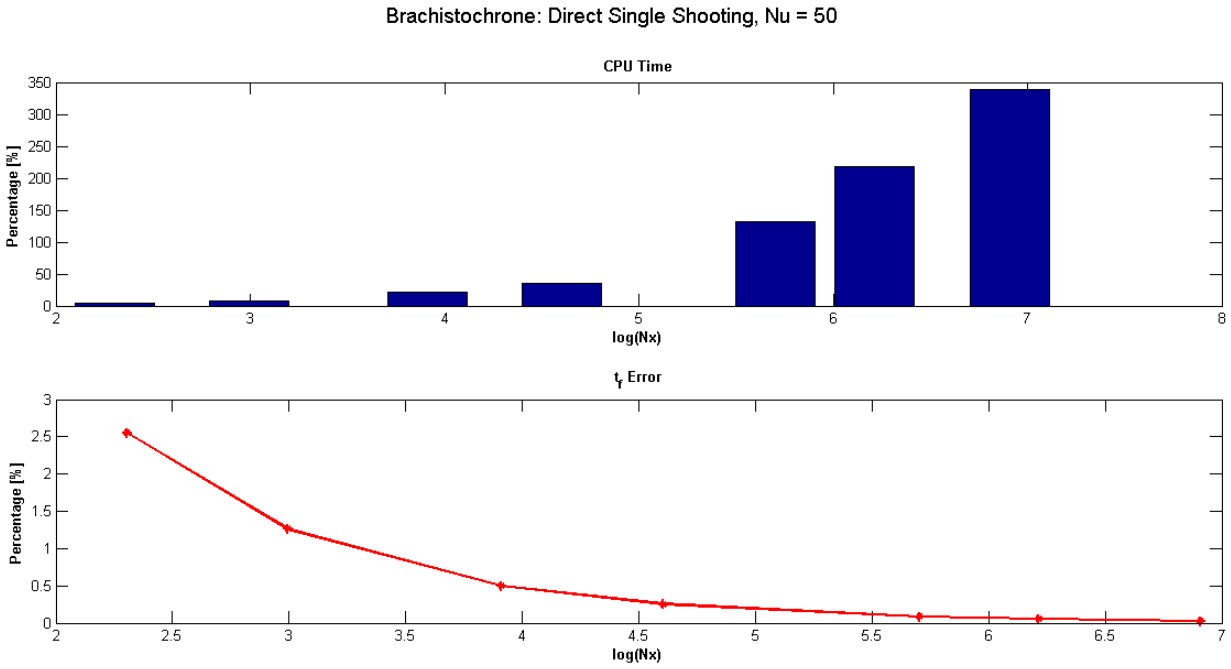

Figure 3.49: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 50$

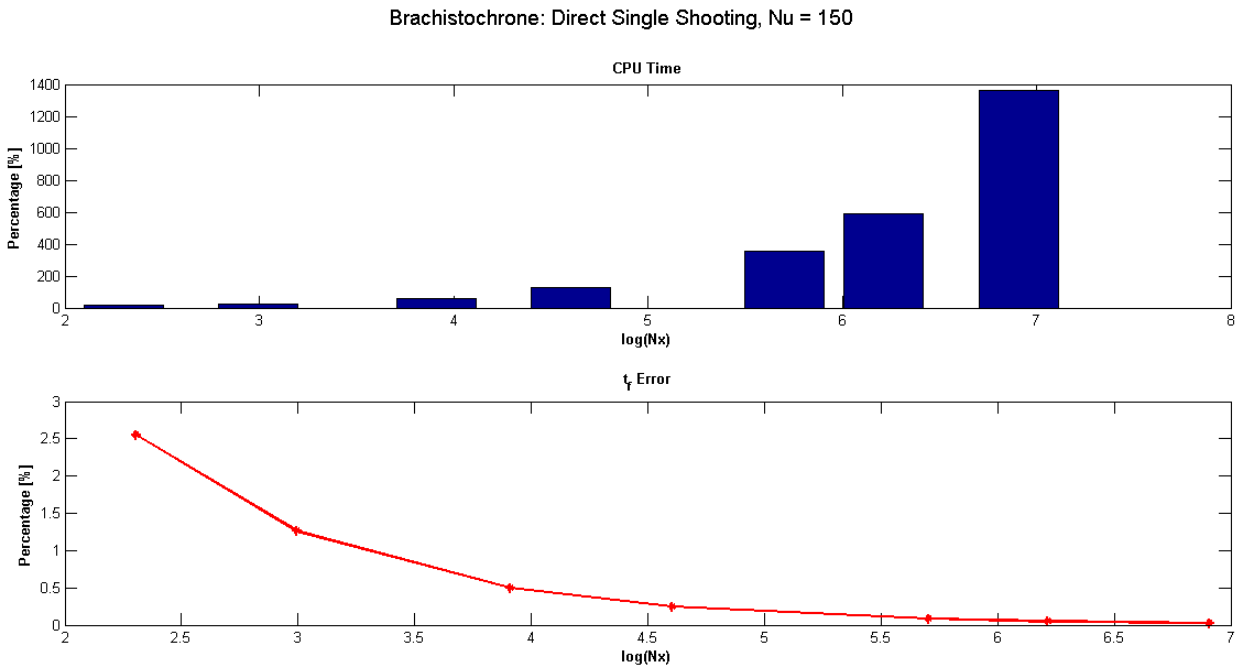

Figure 3.50: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 150$

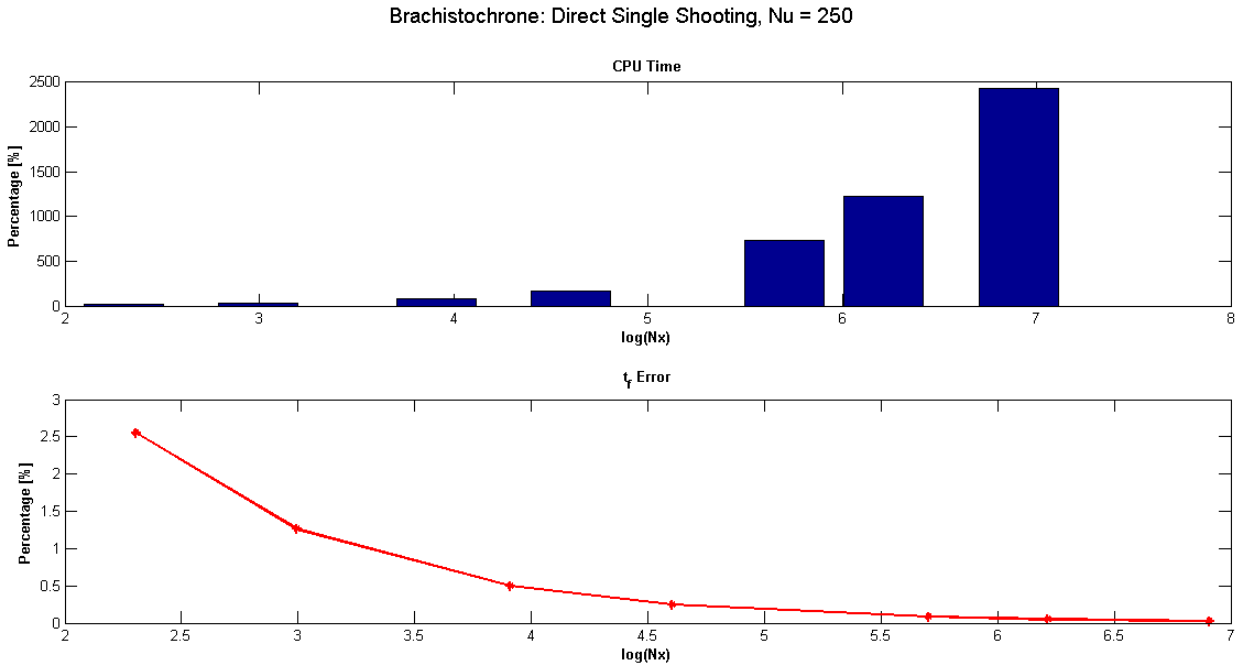

Figure 3.51: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 250$

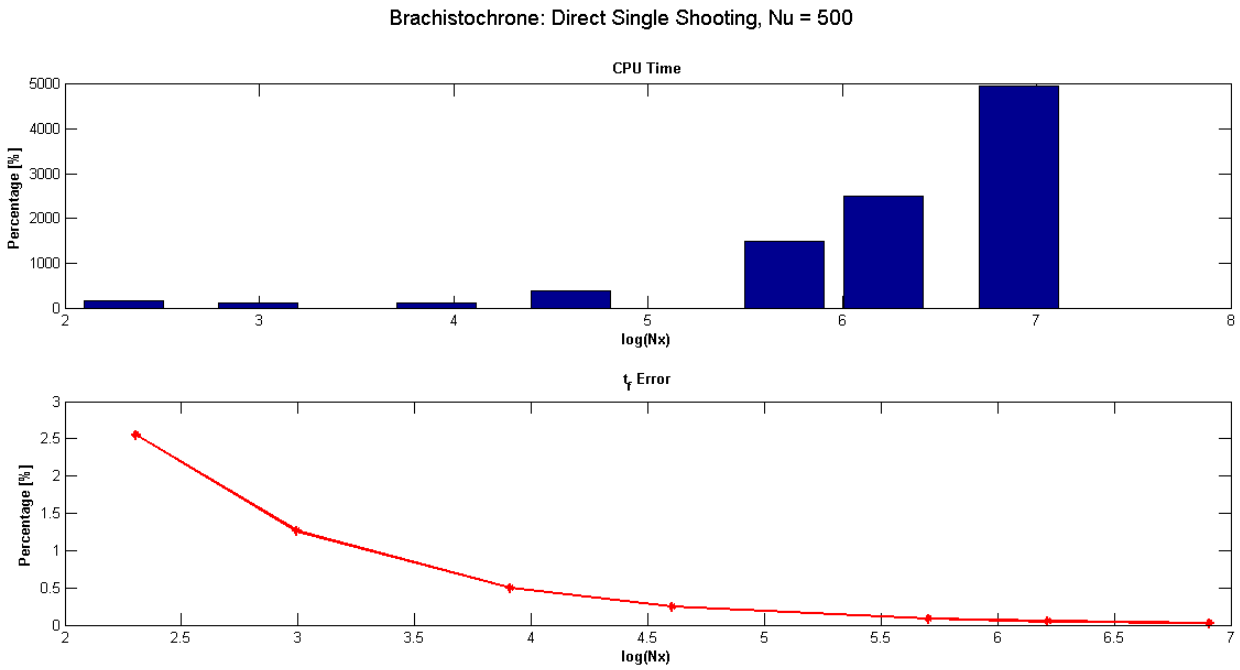

Figure 3.52: Brachistochrone: Direct Single Shooting CPU time and t_f, $N_u = 500$

Table 3.1: Brachistochrone: Direct Single Shooting CPU Times

	Direct Single Shooting - % CPU Times						
	Nu						
Nx	**5**	**10**	**25**	**50**	**150**	**250**	**500**
10	0.45%	1.34%	1.34%	4.02%	17.41%	17.86%	167.41%
20	0.89%	1.79%	4.46%	8.04%	21.88%	28.57%	99.17%
50	2.68%	5.80%	11.16%	20.98%	57.59%	85.27%	104.46%
100	5.80%	10.27%	20.09%	35.27%	129.46%	166.52%	371.88%
300	14.73%	28.57%	58.93%	132.14%	352.23%	736.61%	1501.34%
500	24.55%	48.21%	99.55%	218.75%	586.61%	1227.43%	2494.20%
1000	35.71%	70.54%	197.77%	338.39%	1361.16%	2429.02%	4954.91%

3.4.4.2.2 Direct Multiple Shooting Results The CPU time and t_f error plots for the direct multiple shooting method are given in figures 3.53 to 3.57. Figure 3.53 shows that for $M = 2$ the t_f error decreases for increasing N_u however the correct final time is not achieved and the CPU time taken is almost 3 times the nominal. Overall the results show that as the number of sections increases the error in the final time decreases however the time taken by the multiple

shooting method is much higher than the nominal. For all values of M at least 10 control points are required to reach a percentage error of below 1% and the CPU time increases from 1 to 4000 times more than the nominal case. The CPU percentages for the multiple shooting method are tabulated in table 3.2.

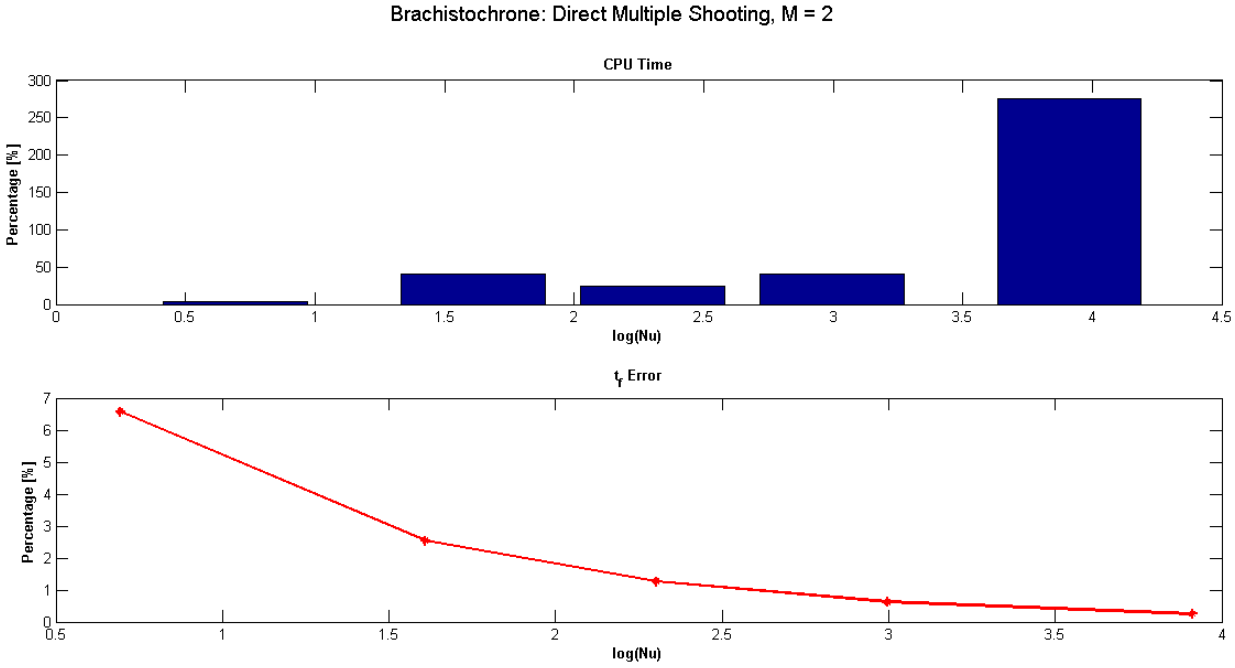

Figure 3.53: Brachistochrone: Direct Multiple Shooting CPU time and t_f, $M = 2$

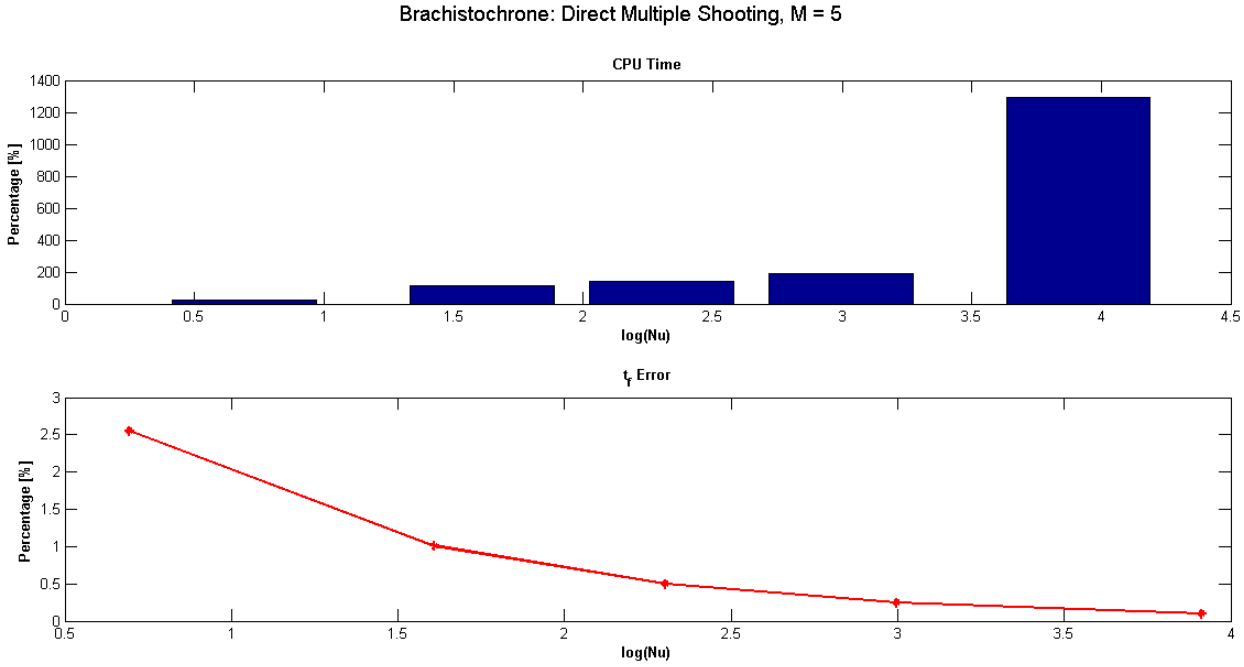

Figure 3.54: Brachistochrone: Direct Multiple Shooting CPU time and t_f, $M = 5$

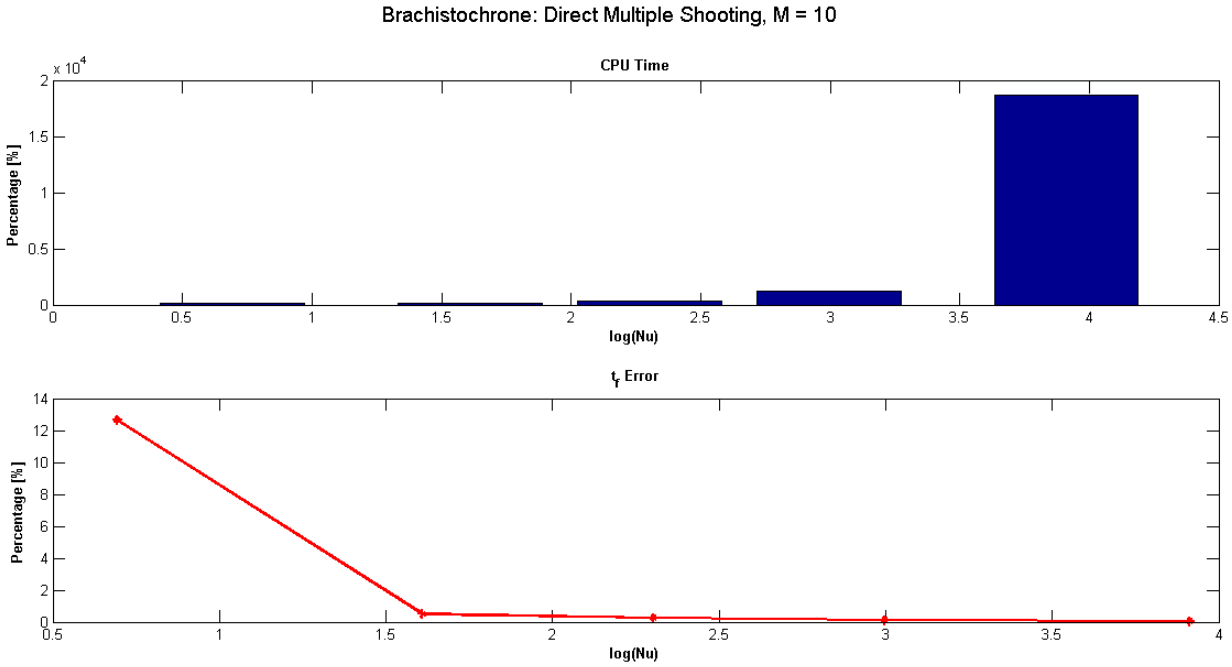

Figure 3.55: Brachistochrone: Direct Multiple Shooting CPU time and t_f, $M = 10$

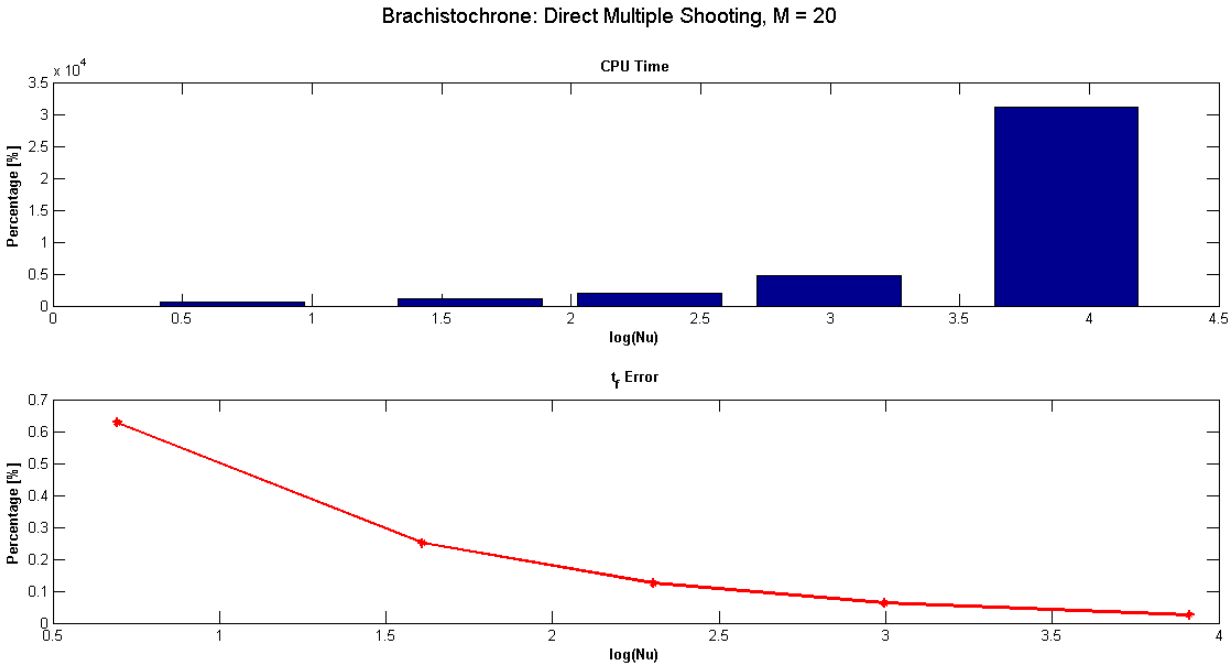

Figure 3.56: Brachistochrone: Direct Multiple Shooting CPU time and t_f, $M = 20$

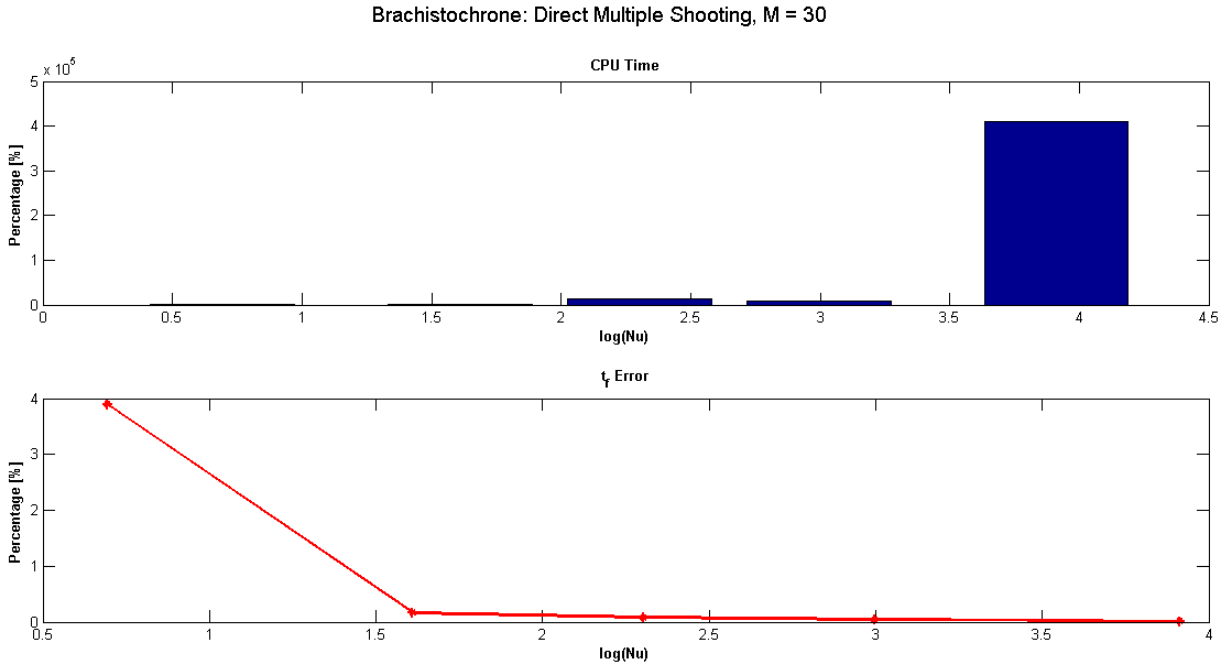

Figure 3.57: Brachistochrone: Direct Multiple Shooting CPU time and t_f, $M = 30$

Table 3.2: Brachistochrone: Direct Multiple Shooting CPU Times

	Direct Multiple Shooting - % CPU Times				
	M				
Nu	**2**	**5**	**10**	**20**	**30**
2	3.57%	21.43%	141.07%	610.27%	1984.38%
5	40.18%	115.18%	149.55%	1184.38%	1598.66%
10	24.11%	143.75%	310.27%	1927.68%	13069.20%
20	40.63%	191.07%	1262.95%	4790.63%	8928.13%
50	275.89%	1292.86%	18788.84%	31184.82%	410166.52%

3.4.4.2.3 Collocation Results - Euler Integration and Pseudospectral The CPU and final time error plots for the collocation Euler integration and pseudospectral methods are provided in figures 3.58 and 3.59 respectively. The accuracy in the final time can be seen to be much higher for both of these methods as compared to both of the shooting methods. The accuracy of the pseudospectral method is higher than for the collocation method. As expected the CPU time increases for both methods as the value of N increases. In general the CPU time taken by the pseudospectral method is higher compared to the collocation method for a given

value of N greater than 10. For $N = 800$ the pseudospectral method takes 14000 times more CPU time than the nominal while the collocation method takes only 1800 times more CPU time (see tables 3.3 and 3.4) . The Pseudospectral method has a percentage error of 0.08% for $N = 5$ compared to the collocation method which is 3 times higher at 0.3% while both take only 1.34% of the nominal CPU time. Clearly the Pseudospectral method can reach a higher level of accuracy with fewer points.

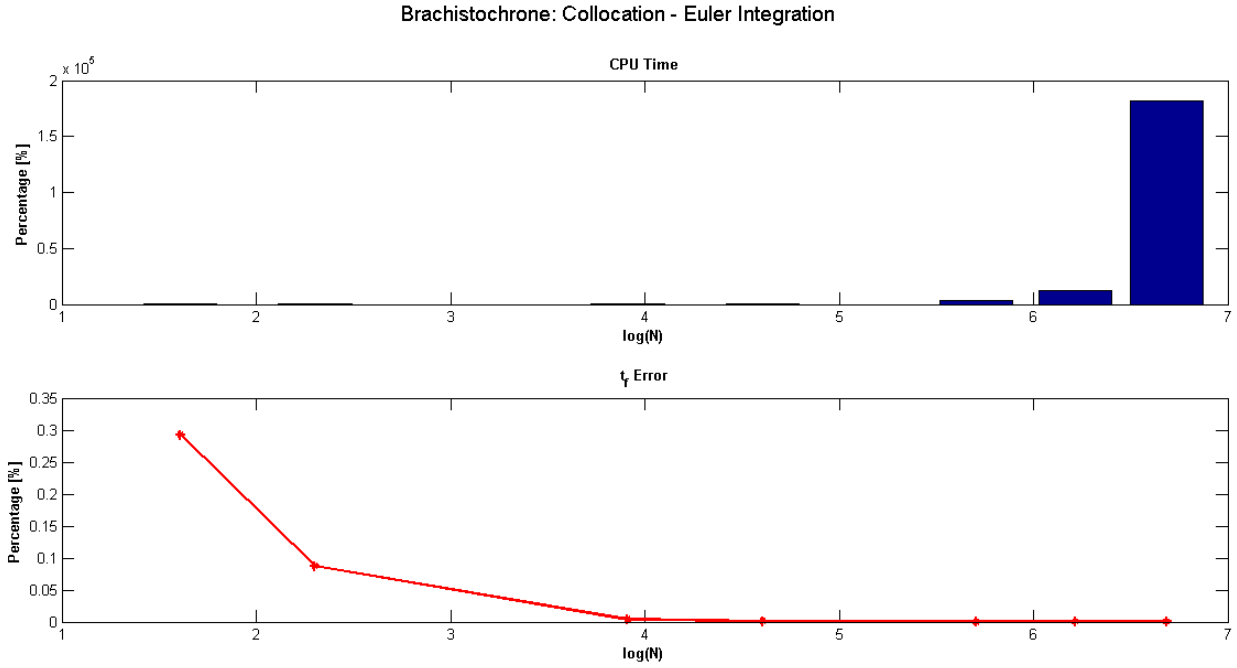

Figure 3.58: Brachistochrone: Collocation - Euler Integration, CPU time and t_f

Table 3.3: Brachistochrone: Collocation - Euler Integration, CPU Times

Collocation, Euler Integration - % CPU Times							
N	**5**	**10**	**50**	**100**	**300**	**500**	**800**
	1.34%	3.13%	37.95%	190.18%	3608.04%	12134.38%	181751.34%

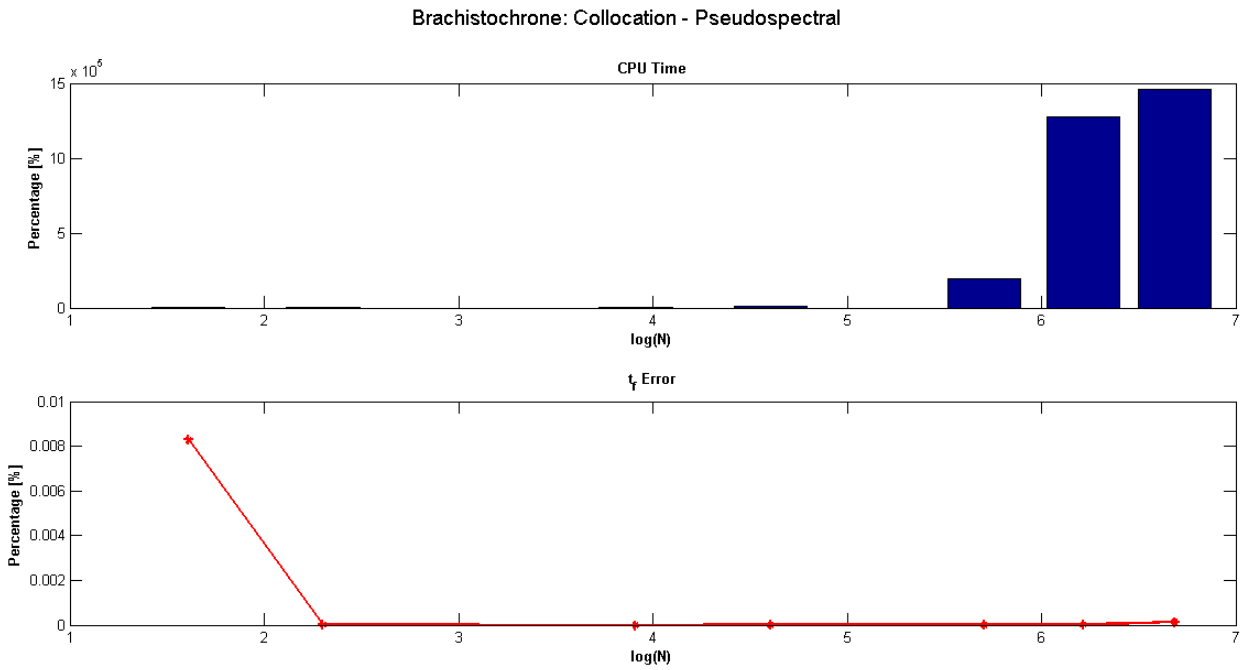

Figure 3.59: Brachistochrone: Collocation - Pseudospectral CPU time and t_f

Table 3.4: Brachistochrone: Collocation - Pseudospectral CPU Times

Pseudospectral - % CPU Times							
N	**5**	**10**	**50**	**100**	**300**	**500**	**800**
	1.34%	3.13%	100.00%	14256.25%	192790.63%	1280342.86%	1468914.29%

3.4.4.3 Findings and Conclusion

The results show that the pseusdospectral method can produce more accurate results with fewer discretisation points consequently requiring less time. While, for large values of N, the pseudospectral method results in a longer CPU time, larger values of N are deemed unnecessary to obtain a high level of accuracy. For this reason I use only the Pseudospectral method with $N = 50$ points in the implementation of NMPC for the remainder of this research.

3.5 Illustrative Example: 2D Robot Model

Before designing an MPC based fault tolerant flight controller for an unmanned aerial vehicle (UAV) it is important to gain a thorough understanding of the fundamental workings of MPC. To do this a simple 2D Robot Model is used for controller design and implementation as a simpler

model enables the study of the proposed model and allows for trouble shooting. The focus here is the implementation of the pseudospectral method to NMPC, to recognise its strengths and weaknesses. I also make comparisons of my NMPC solution with linear MPC in the solution of both the open and closed loop control problems.

3.5.1 Equations of motion

Before the model predictive controller can be implemented it is necessary to develop the robot model. The 2D robot model given in figure 3.60 is used for both the linear and nonlinear implementations of MPC.

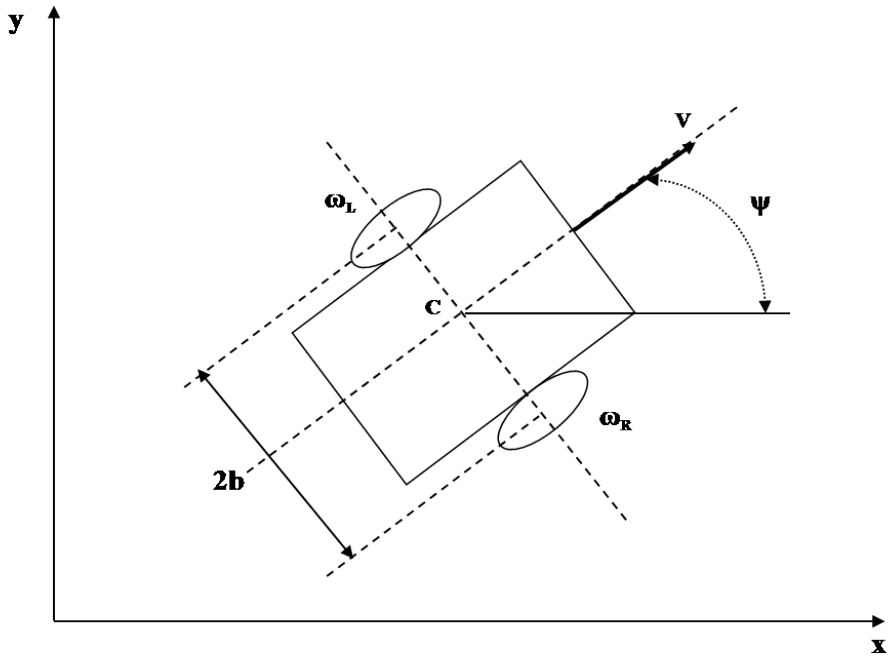

Figure 3.60: Robot Schematic

The relevant equations for this robot model are:

$$\dot{x} = V\cos\psi, \tag{3.51}$$

$$\dot{y} = V\sin\psi, \tag{3.52}$$

$$\dot{\psi} = \frac{R(\omega_R - \omega_L)}{2b}, \tag{3.53}$$

where:

x x-coordinate of the point C,

y y-coordinate of the point C,

ψ heading angle,

ω_R right wheel angular velocity,

ω_L left wheel angular velocity,

V Speed and is given by:

$$V = \frac{R(\omega_R + \omega_L)}{2}. \tag{3.54}$$

The next few sections detail the development of the linear and nonlinear MPC controllers. For a fair comparison the pseudospectral method with 50 collocation points was chosen as the method for discretisation.

3.5.2 Linear MPC

The linear MPC problem is formulated using a linear process model, with linear constraints and a quadratic cost function. The problem is convex with only one global minimum. A state space representation has been chosen to implement the linear controller:

$$\dot{x}(t) = A(t)\, x(t) + B(t)\, u(t). \tag{3.55}$$

The model given in section 3.5.1 clearly shows that the state equations for the the 2D robot are nonlinear and have the general form:

$$\dot{x}(t) = f[x(t), u(t), t]. \tag{3.56}$$

In this work a linear representation of the state equations is used. Hence the nonlinear state equations given in (3.56) must be linearised before they can be converted to state space form. To linearise the state equations we linearise around a nominal solution, $\{\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot)\}$:

$$\mathbf{x}(t) = \mathbf{x}_0(t) + \delta\mathbf{x}(t), \qquad \mathbf{u}(t) = \mathbf{u}_0(t) + \delta\mathbf{u}(t). \tag{3.57}$$

Now assuming that $f(\cdot)$ is smooth and can be represented using a Taylor series expansion:

$$f[x(t), u(t), t] = f[x_0(t), u_0(t), t] + f_x(t)\,\delta x(t) + f_u(t)\,\delta u(t) + o(\delta x, \delta t). \tag{3.58}$$

Where:

$$f_x = \left.\frac{\partial f}{\partial x}\right|_{x_0,u_0}, \qquad f_u = \left.\frac{\partial f}{\partial u}\right|_{x_0,u_0}. \tag{3.59}$$

After applying the Taylor series expansion the linearised equations become:

$$\dot{\delta} = f_x \delta x + f_u \delta u. \tag{3.60}$$

Note that equation (3.60) is in the form of the general state space representation given by equation (3.55), where the A matrix is the matrix of partial derivatives with respect to the states, f_x and the B matrix is a matrix of partial derivatives with respect to the inputs, f_u. These partial derivative matrices are known as Jacobian matrices. The state and input vectors here are no longer the state and inputs themselves but are the perturbed states and inputs, δx and δu respectively. Perturbed states are defined as the difference between the actual states and the nominal solution. Hence a nonlinear system is linearised by linearising the system around a nominal trajectory and the resulting system is a linear system in the perturbed states and not in the states per se. Linear techniques can then be applied. One major drawback, however, is that the linear solution is only feasible within a small region around the point of linearisation.

For our robot system the linearised model is:

$$\begin{bmatrix} \delta\dot{x} \\ \delta\dot{y} \\ \delta\dot{\psi} \end{bmatrix} = A(t) \begin{bmatrix} \delta x \\ \delta y \\ \delta\psi \end{bmatrix} + B(t) \begin{bmatrix} \delta\omega_R \\ \delta\omega_L \end{bmatrix}, \tag{3.61}$$

where the perturbed states are δx, δy and $\delta\psi$ and the perturbed inputs are $\delta\omega_R$ and $\delta\omega_L$. The Jacobian matrices A and B were found to be:

$$A = \begin{bmatrix} 0 & 0 & -V\sin\psi \\ 0 & 0 & -V\cos\psi \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} \frac{R}{2}\cos\psi & \frac{R}{2}\cos\psi \\ \frac{R}{2}\sin\psi & \frac{R}{2}\sin\psi \\ \frac{R}{2b} & -\frac{R}{2b} \end{bmatrix}. \tag{3.62}$$

Note that this is a time varying system, hence the values of the A and B matrices are recalculated at each time step.

In my implementation of linear MPC, pseudospectral discretisation will be used. Hence, given N collocation points the optimization state vector $\mathbf{x}$ will be of length $(n_x + n_u)(N+1)$, where n_x

and n_u are the number of states and inputs respectively. The aim is to have the robot follow a pre-specified or "nominal" (also referred to as reference) trajectory. The states to be minimised are the perturbed states, i.e. the difference between the actual and nominal. The system is subject to the state dynamics, upper and lower bounds on the state, and initial and terminal constraints:

$$J_N = \mathcal{M}\left[\mathbf{x}_N\right] + \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \mathcal{L}\left[\mathbf{x}_j, \mathbf{u}_j, t_j\right] w_j, \tag{3.63}$$

subject to:

$$\left(\frac{t_f - t_0}{2}\right) \mathbf{D}_{j,k}\, \delta\mathbf{x}\left(t\right) - \left(A\left(t\right) \delta\mathbf{x}\left(t\right) + B\left(t\right) \delta\mathbf{u}\left(t\right)\right) = 0, \tag{3.64}$$

$$\delta\mathbf{x}_i = \mathbf{x}(t_0) - \mathbf{x}_{\text{ref}}(t_0), \tag{3.65}$$

$$\delta\mathbf{x_f} = 0, \tag{3.66}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{3.67}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}. \tag{3.68}$$

As the system is time varying the A and B matrices are calculated at every time step for each collocation point over the prediction horizon.

The linear MPC process works as follows: firstly the initial and terminal constraints are set. The initial constraint is set to the difference between the actual state at time t_0 and the reference trajectory at time t_0. The terminal constraint is set to zero, as it is desired to have a perturbation of zero by the end of the prediction horizon. This is followed by calculations of the nominal inputs $u_0 = [\omega_{R0}, \omega_{L0}]^\top$. These nominal values represent the control inputs required to achieve the desired reference trajectory if the plant was perfectly situated on this trajectory. The Jacobian matrices are then calculated using these nominal values for states and inputs. During each sampling interval these calculations are performed for each collocation point along the prediction horizon. Once all of the equality constraints are set up the SNOPT [54] optimisation procedure is called on to calculate the optimal δu which minimises the cost function while satisfying all constraints. A vector of length $(n_x + n_u)(N + 1)$ will be produced with each solution corresponding to a collocation point. Only the first input from this vector is applied to the plant. The optimisation vector is a vector of perturbed states hence the actual control input which is to be applied to the plant must be calculated via:

$$u = u_0 + \delta u. \tag{3.69}$$

The control input is applied to the plant which is then simulated forward in time, and the whole procedure repeats until the end of the simulation time.

3.5.3 Nonlinear MPC

The underlying concept of the nonlinear MPC controller is the same as its linear counterpart. That is, an optimal control problem is solved over a finite prediction horizon, only the first control input is applied to the plant, the prediction window slides along by the sampling time interval and the whole procedure is repeated. The main difference is that the state vector comprises actual states rather than the perturbed states. The cost function to be minimised is:

$$J_N = \mathcal{M}\left[\mathbf{x}_N\right] + \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \mathcal{L}\left[\mathbf{x}_j, \mathbf{u}_j, t_j\right] w_j, \tag{3.70}$$

subject to:

$$\left(\frac{t_f - t_0}{2}\right) \mathbf{D}_{j,k}\mathbf{x}\left(t\right) - \dot{\mathbf{x}} = 0, \tag{3.71}$$

$$\mathbf{x}(t_0) - \mathbf{x}_{\text{ref}}(t_0) = 0, \tag{3.72}$$

$$\mathbf{x}(t_f) - \mathbf{x}_{\text{ref}}(t_f) = 0, \tag{3.73}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{3.74}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}. \tag{3.75}$$

The next section describes the results obtained by applying the controllers to an open loop problem.

3.5.4 The Open Loop Problem

In MPC an open loop problem is solved at each time step. Hence before NMPC can be applied to fault tolerant control it was important to understand the open loop problem. This provides an insight into the factors that affect the solution.

There are many tuning parameters used to determine the performance of the controller; the weighting factors on the cost function, the design of the cost function, the length of the prediction horizon, the initial condition, the integration time step, and the number of discretisation points required for an acceptable solution. From the previous analysis the numerical technique chosen for the application of NMPC is the Pseudospectral method with 50 discretisation/coincidence points. This section looks at the effect of the design of the cost function, the prediction window

length, the integration time step and the affect of the initial condition on the solution.

The 2D robot is required to follow the given path:

$$\forall\, x \geq 0 : y = 5, \tag{3.76}$$

travelling with a velocity of $1m/s$ and constraints of ± 1000deg/sec on the wheel speeds ω_R and ω_L.

Both linear and nonlinear MPC methods were applied. The state and input vector for the linear MPC controller are:

$$\mathbf{x}_{\text{lin}} = [\delta x \;\; \delta y \;\; \delta\psi]^\intercal, \tag{3.77}$$

$$\mathbf{u}_{\text{lin}} = [\delta\omega_R \;\; \delta\omega_R]^\intercal. \tag{3.78}$$

The state and input vector for the nonlinear MPC controller are:

$$\mathbf{x}_{\text{nl}} = [x \;\; y \;\; \psi]^\intercal, \tag{3.79}$$

$$\mathbf{u}_{\text{nl}} = [\omega_R \;\; \omega_L]^\intercal. \tag{3.80}$$

The overall objective is to drive the robot back to the reference path from $y = 6$ to $y = 5$.

3.5.4.1 Effect of Different Cost Functions

I now look at the effect of selecting a cost function on the MPC solution. Five different cost functions were developed:

Cost Type 1: Errors between the reference/nominal path and the robot path are minimised:

$$J_{N1} = \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \left(\left\| \mathbf{x} - \mathbf{x}_{\text{ref}} \right\|^2_{Q_x} \right) w_j. \tag{3.81}$$

Cost Type 2: Errors between the robot path and the nominal path, plus the error between the actual wheel speeds, ω_R and ω_L and the nominal wheel speeds are minimised:

$$J_{N2} = \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \left(\left\| \mathbf{x} - \mathbf{x}_{\text{ref}} \right\|_{Q_x}^2 + \left\| \mathbf{u} - \mathbf{u}_{\text{ref}} \right\|_{Q_u}^2 \right) w_j. \tag{3.82}$$

Cost Type 3: Errors between the robot path and the nominal path, plus the difference between the wheel speeds are minimised:

$$J_{N3} = \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \left(\left\| \mathbf{x} - \mathbf{x}_{\text{ref}} \right\|_{Q_x}^2 + \left\| \omega_R - \omega_L \right\|_{Q_\omega}^2 \right) w_j. \tag{3.83}$$

Cost Type 4: Errors between the nominal speed and robot speed as well as the errors between the nominal angular acceleration and the robot's angular acceleration are minimised:

$$J_{N4} = \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \left(\left\| V - V_{\text{ref}} \right\|_{Q_V}^2 + \left\| \dot{\psi} - \dot{\psi}_{\text{ref}} \right\|_{Q_\psi}^2 \right) w_j. \tag{3.84}$$

Cost Type 5: Errors between the nominal speed and robot speed as well as the errors between the nominal angular acceleration and the robot's angular acceleration along with the errors between the nominal path and the robot path are minimised:

$$J_{N5} = \frac{(t_f - t_0)}{2} \sum_{j=0}^{N} \left(\left\| \mathbf{x} - \mathbf{x}_{\text{ref}} \right\|_{Q_x}^2 + \left\| V - V_{\text{ref}} \right\|_{Q_V}^2 + \left\| \dot{\psi} - \dot{\psi}_{\text{ref}} \right\|_{Q_\psi}^2 \right) w_j. \tag{3.85}$$

Each cost type was tested using both linear and nonlinear MPC. The robot initial x, y and ψ was set to $\mathbf{x}_0 = [0\,6\,0]^\intercal$ for both controllers. The prediction window length was also varied and was equal to $H_p = 1\,\text{sec}$, $H_p = 5\,\text{secs}$ and $H_p = 10\,\text{secs}$. The cost function weights are square matrices with the following diagonal values for each state: $Q_x = 10$, $Q_u = 1$, $Q_\psi = 1$, $Q_{\dot{\psi}} = 1$ and $Q_V = 1$.

The optimal trajectories produced by all the different cost functions for a prediction window length of 1 sec are given in figures 3.61 and 3.62 for linear and nonlinear MPC respectively. Cost types 1, 2, 3 and 5 were able to drive the robot back onto the desired path by the end of the window for both the linear and nonlinear cases, however cost type 4 was unsuccessful in doing so.

When the window length is increased to 5 seconds the optimal trajectories are seen to be much smoother in driving the robot back to the path compared to the 1 second window length. Figures 3.63 and 3.64 show the results for the linear and nonlinear controllers respectively for a window length of 5 seconds. All cost function types were able to drive the robot back to the path by the end of the window length in both linear and nonlinear cases. Cost type 1 took longer to bring the robot back to the path with a linear controller when compared to the nonlinear controller. The nonlinear controller drove the robot straight towards the path while the linear controller brought the robot back smoothly. The nonlinear controller for cost type 3 initially drove the robot away from the path before bringing it back on track while cost type 4 only brought the robot back right at the end of the prediction window. This was not the case with the linear controller, with the linear controller cost type 3 driving the robot back to the path. Cost types 2 and 5 behaved similarly for both controllers in that they both brought the robot back to the path along a smooth path. However cost type 5 slightly overshot the path before bringing the robot back on track with the linear controller.

Increasing the prediction window length further to a 10 second look ahead shows that all cost types with both linear and nonlinear controllers were able to drive the robot back to the path. The nonlinear controller (figure 3.66) was able to bring the robot back within approximately 2 seconds for all cost types, except cost 1 which brought the robot back smoothly in approximately 0.5 secs and cost type 4 which only reached the path at the end of the window length. The linear controller results show that the robot was back on track for all cost types by approximately 4.5 secs, other than cost type 1 which drove the robot to the path in 1.5 secs.

The error plots for all the prediction window lengths are given in figures 3.67 through to 3.69. The plots show the magnitude of the error in the y-direction between the nominal path ($y = 5$) and the actual robot path. The results show that for all cost types and all window lengths the nonlinear controller produced lower error compared to the linear controller. The error between the two controllers decreases as the robot approaches the path. The results show that as the perturbation decreases the linear controllers performs equally as well as the nonlinear controller, however for larger perturbations the nonlinear outperforms the linear controller. This is true for all cost types except cost type 4. Cost type 4 results in higher errors compared to the linear controller across all prediction window lengths and the nonlinear controller is only able to drive the robot to the path at the end of the window, obeying the terminal constraint. Cost type 4

is the only cost function which does not minimise the path errors instead minimising velocity and angular acceleration only. The linear controller produces lower errors as the states are the perturbed states from the nominal. Cost type 1 in the nonlinear case, brings the robot back to the path in the least amount of time however the solution oscillates around the nominal path once the path has been reached as is evident from figure 3.66. The results show that for a path following scenario it is best to not only minimise the path errors but to also follow a velocity profile to obtain a smoother non oscillating solution. For this reason I will continue the remainder of this research using cost type 5 as the cost function of choice. In addition, the error plots show that the difference in errors produced by the linear and nonlinear controllers are the least for this cost function making it the best candidate for comparison purposes.

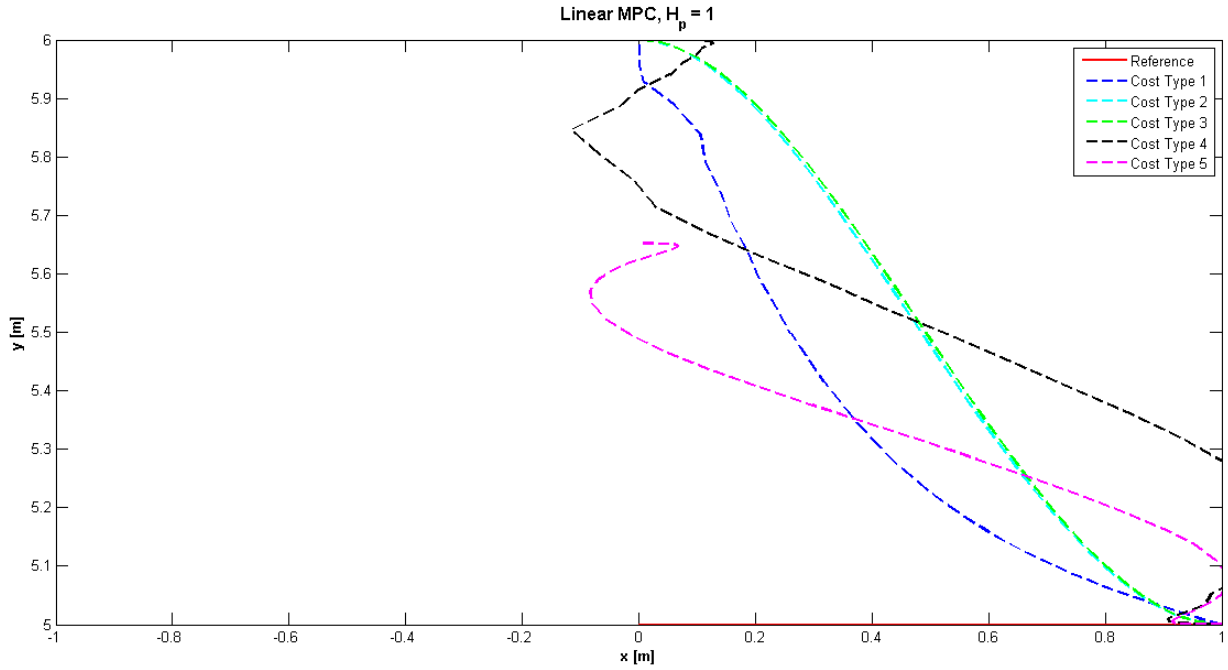

Figure 3.61: Open Loop: Different Cost Functions, $H_p = 1$, Linear MPC

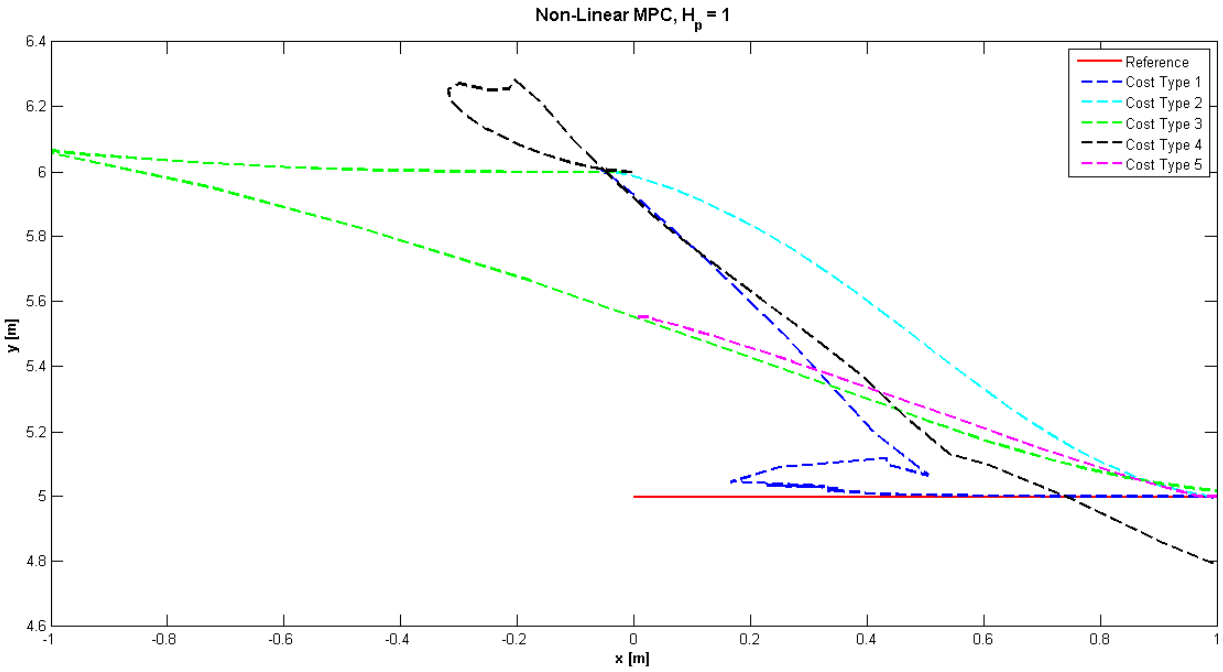

Figure 3.62: Open Loop: Different Cost Functions, $H_p = 1$, Nonlinear MPC

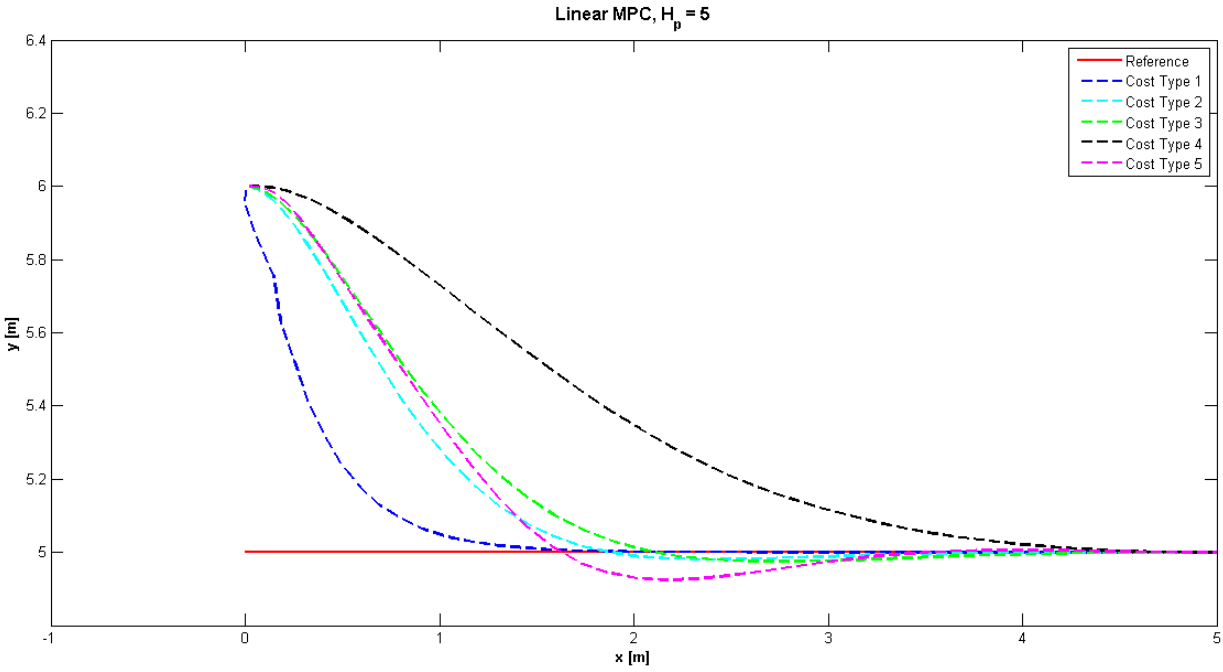

Figure 3.63: Open Loop: Different Cost Functions, $H_p = 5$, Linear MPC

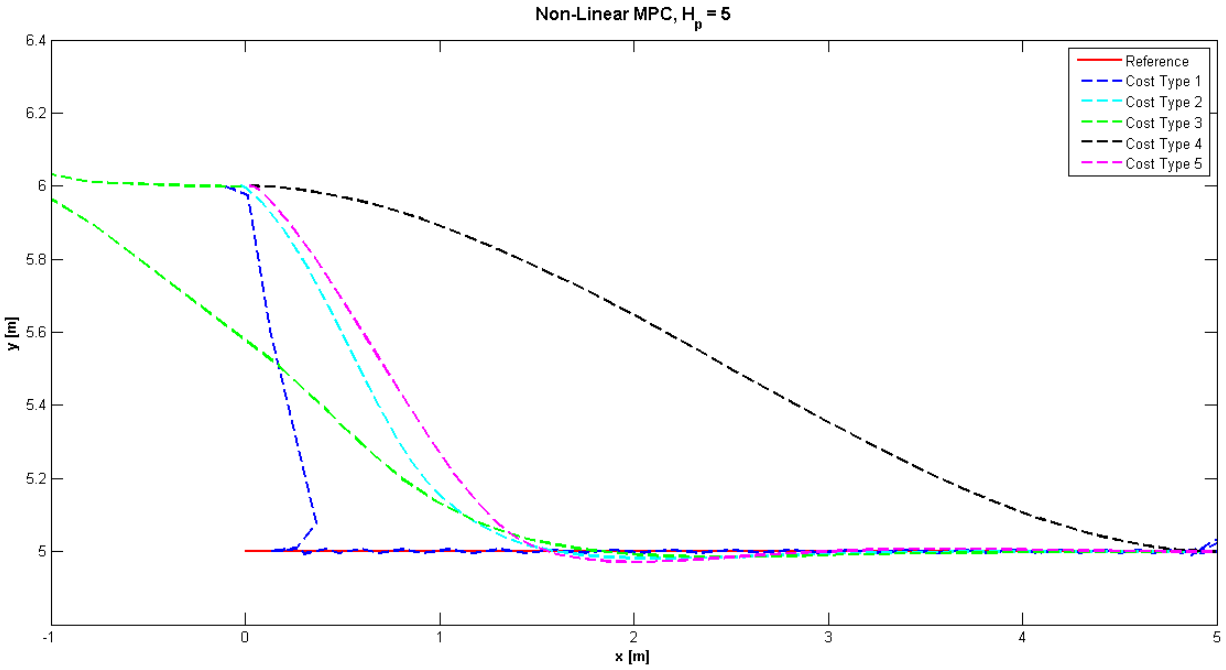

Figure 3.64: Open Loop: Different Cost Functions, $H_p = 5$, Nonlinear MPC

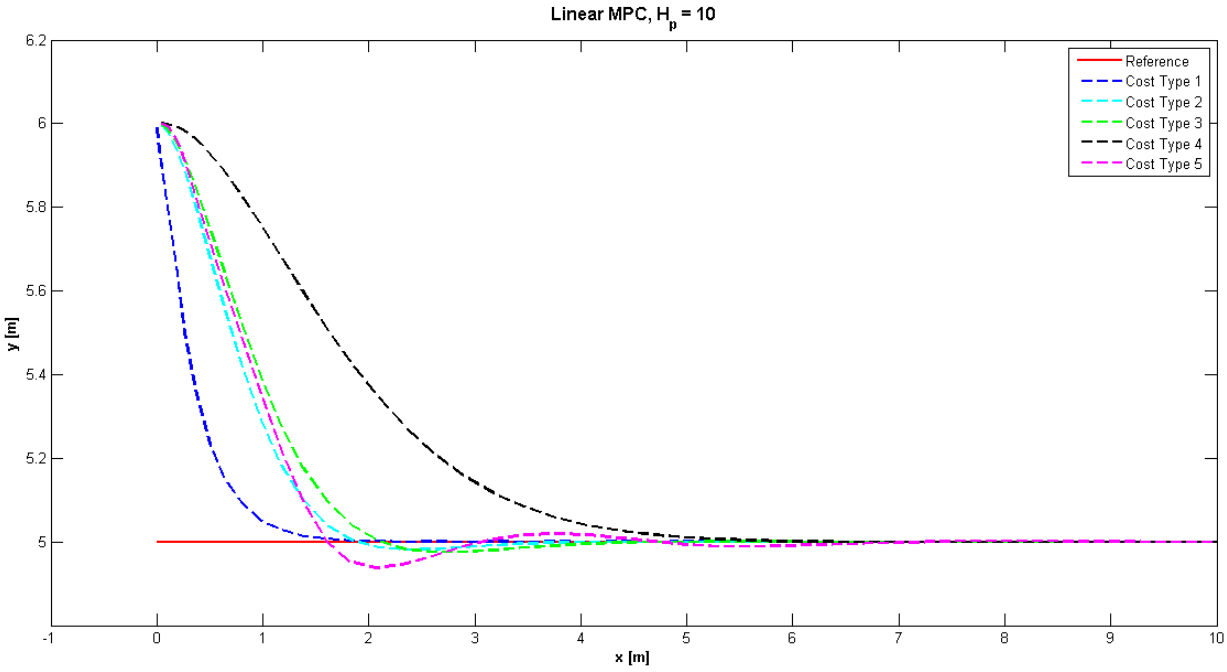

Figure 3.65: Open Loop: Different Cost Functions, $H_p = 10$, Linear MPC

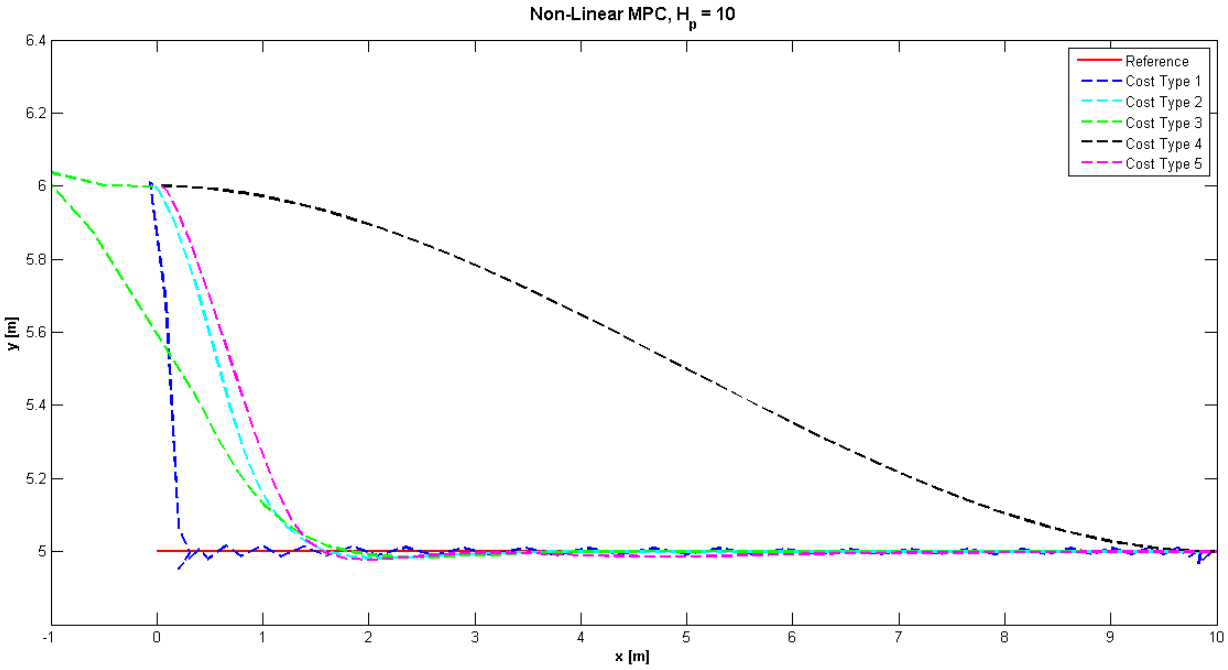

Figure 3.66: Open Loop: Different Cost Functions, $H_p = 10$, Nonlinear MPC

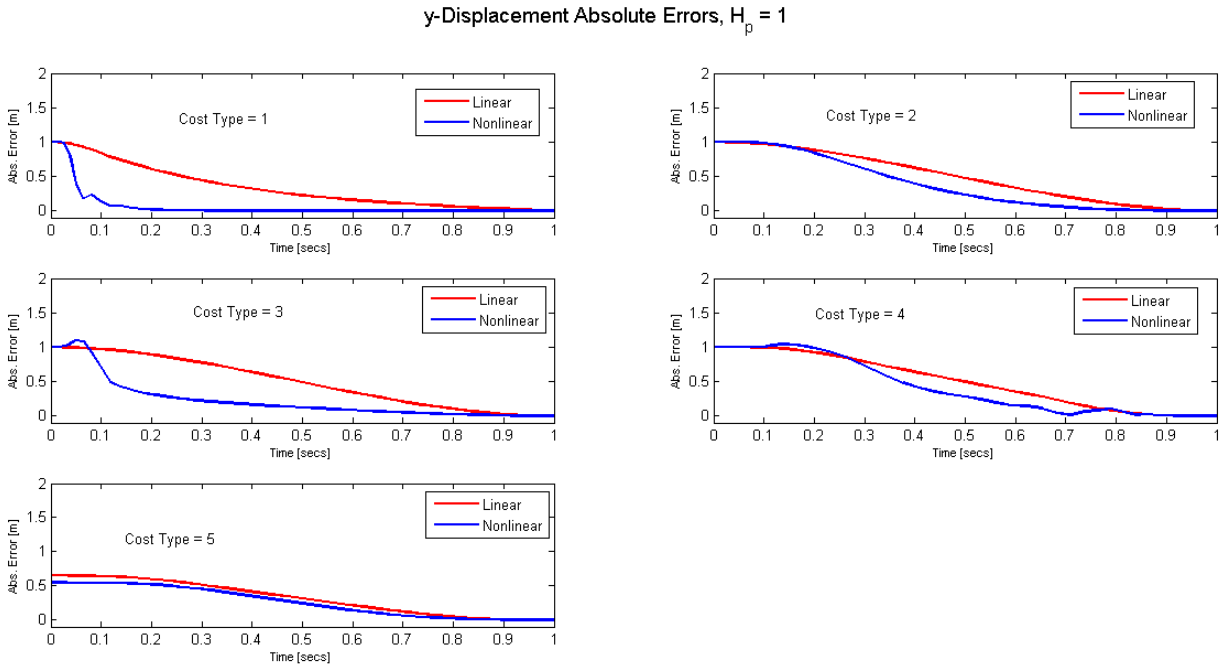

Figure 3.67: Open Loop: Different Costs y-Displacement Errors, $H_p = 1$

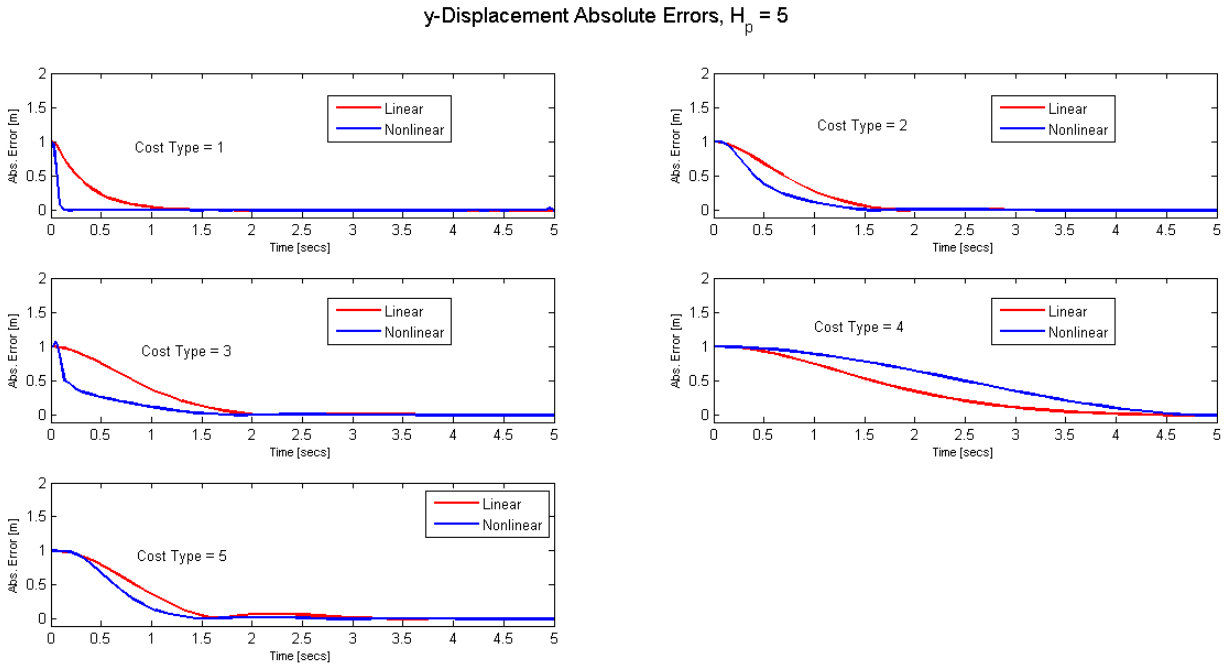

Figure 3.68: Open Loop: Different Costs y-Displacement Errors, $H_p = 5$

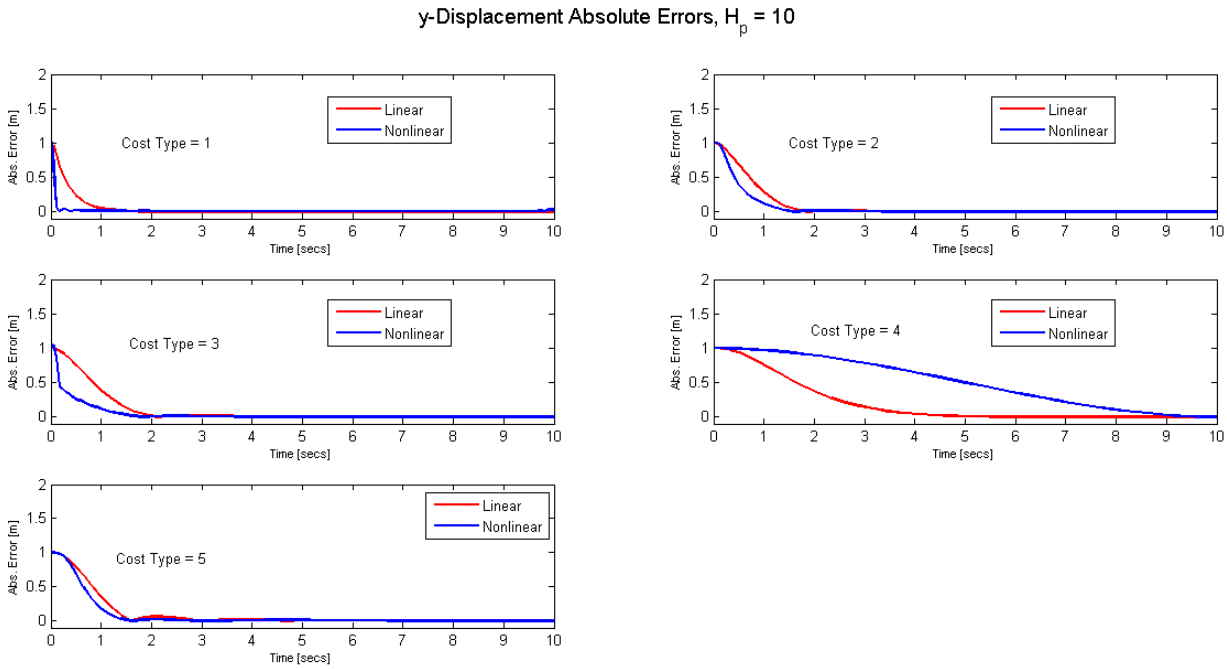

Figure 3.69: Open Loop: Different Costs y-Displacement Errors, $H_p = 10$

3.5.4.2 Effect of the Integration Time Step

Sub-section 3.5.4.1 investigated the effect of different cost functions on the MPC solution. This sub-section looks at the effect the integration time step has on the overall solution. Up until

now only the optimal solution produced by the controller was investigated, however in an MPC framework only the first output is applied to the plant, with the plant then providing sensor information to the navigation subsystem, for example, (on an aircraft) to calculate location and orientation information. Hence its is important to understand the effect of the the integration time step in conjunction with the optimal control input.

The integration time step was varied as follows:

$$dt = [0.1,\ \ 0.01,\ \ 0.001],$$

for varying H_p lengths namely 1 sec, 5 secs and 10 secs. The results are given in figures 3.70 to 3.75. All results show that the integration time step has very little effect on the results. When zooming in on the results the smallest integration time step of 0.001secs was shown to give a solution closest to the optimal. The length of the prediction horizon was seen to have the greatest effect on the integrated solution. The longer the look ahead the closer the integrated solution is to the optimal solution. Another point to note is that the integrated solution produced with the nonlinear controller more closely matches the optimal solution compared to the linear solution.

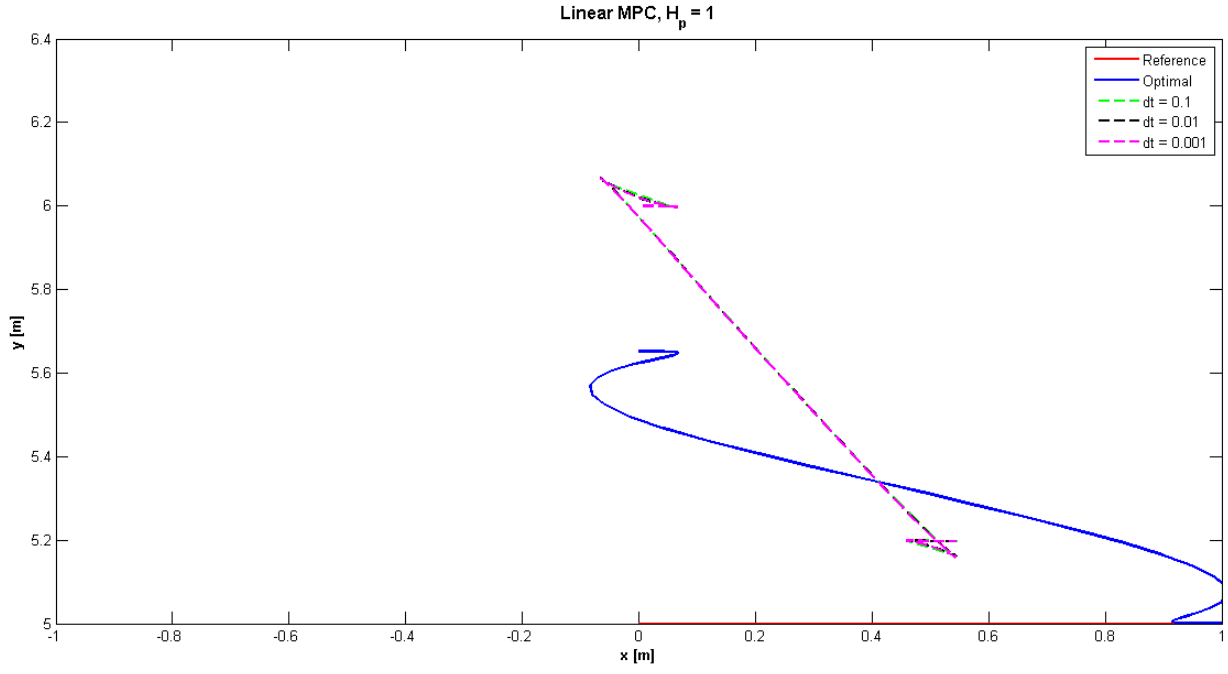

Figure 3.70: Open Loop: Different dt, $H_p = 1$ sec, Linear MPC

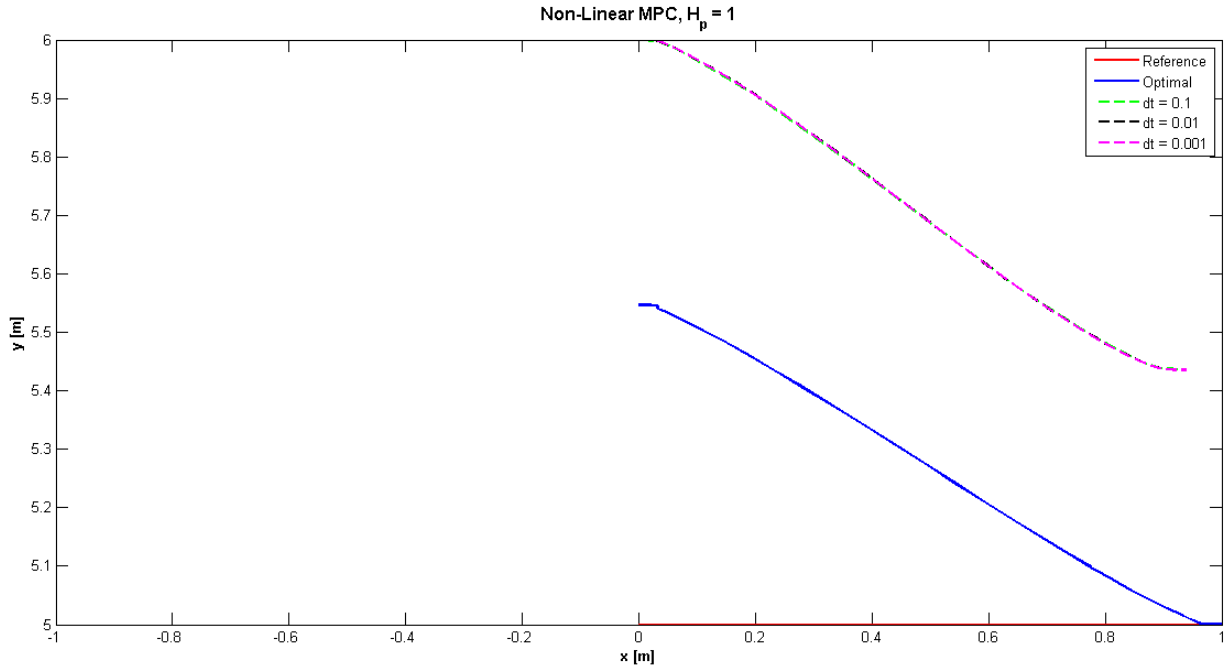

Figure 3.71: Open Loop: Different dt, $H_p = 1$ sec, Nonlinear MPC

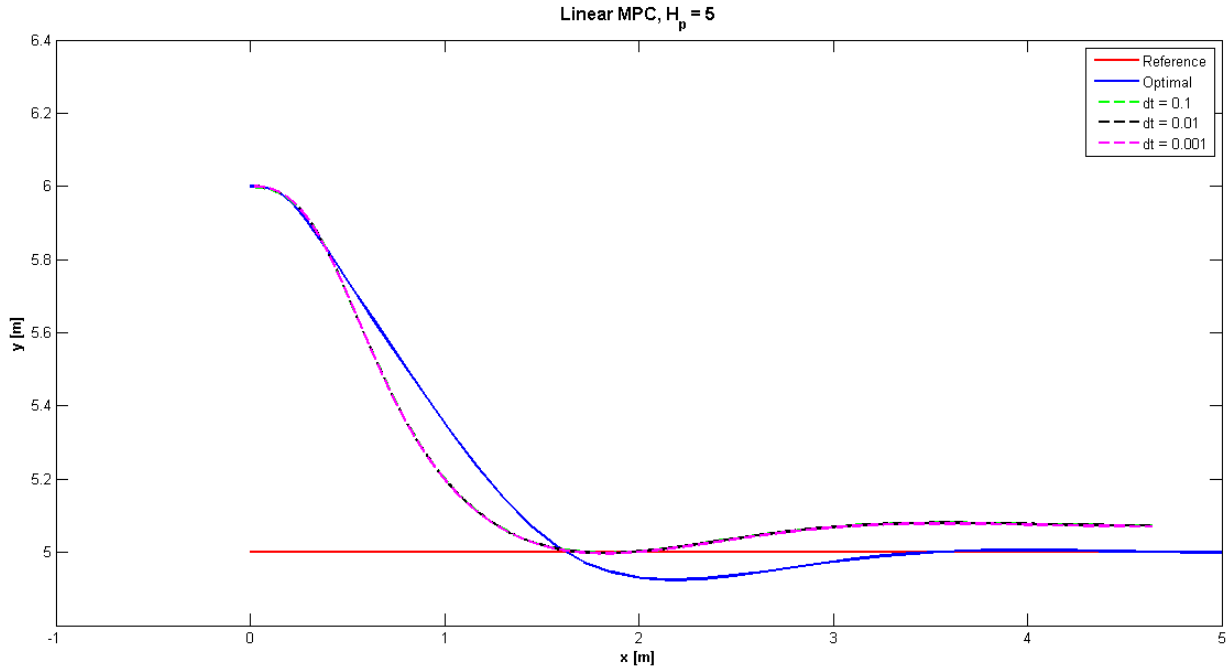

Figure 3.72: Open Loop: Different dt, $H_p = 5$ secs, Linear MPC

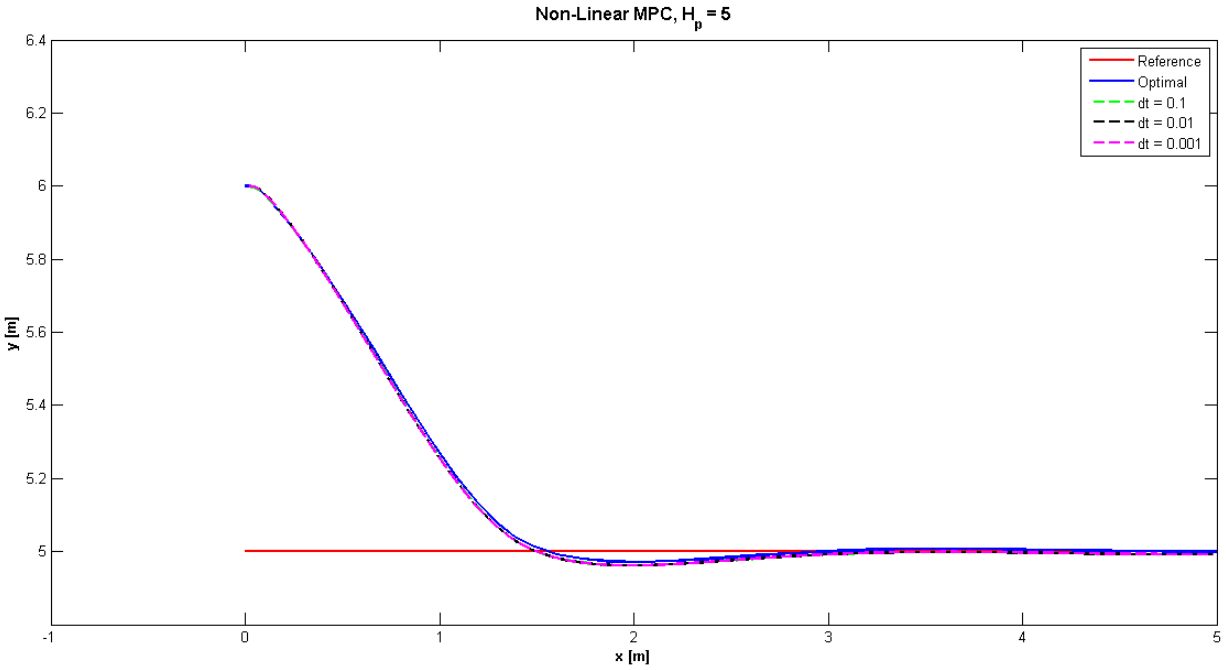

Figure 3.73: Open Loop: Different dt, $H_p = 5$ secs, Nonlinear MPC

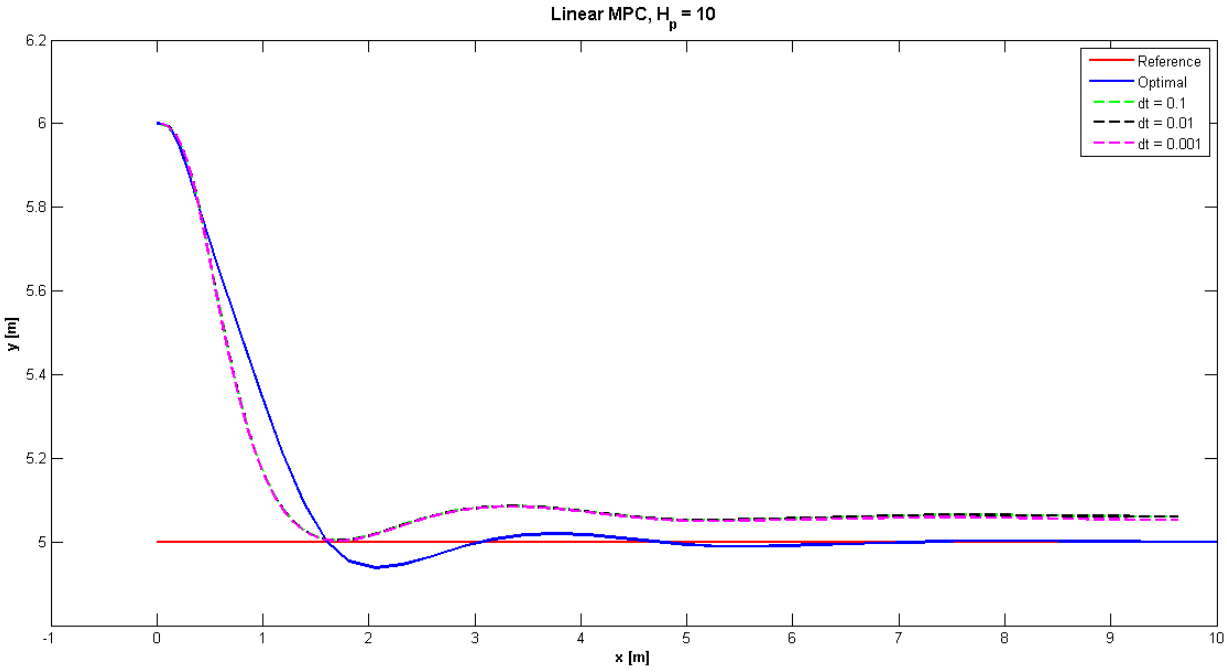

Figure 3.74: Open Loop: Different dt, $H_p = 10$ secs, Linear MPC

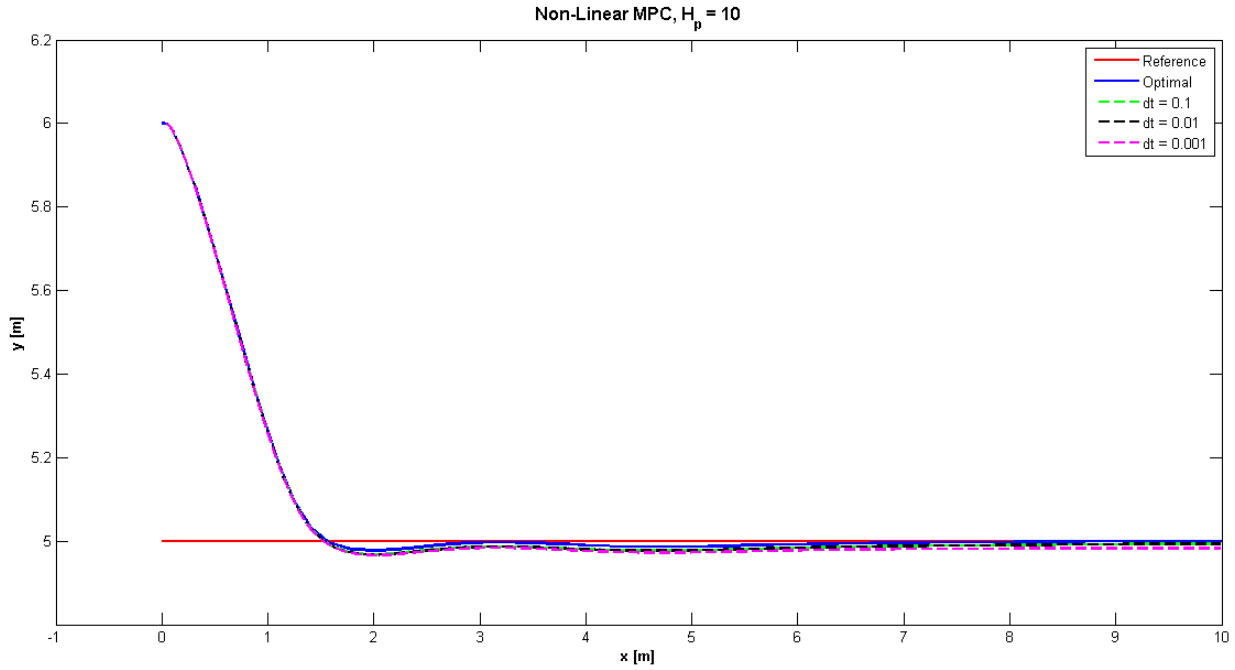

Figure 3.75: Open Loop: Different dt, $H_p = 10$ secs, Nonlinear MPC

Varying the length of the prediction window showed that it is always best to have a longer window. The longer window length produced the lowest errors particularly in the linear controller case. In addition, the longer window allowed the robot to reach the nominal path quicker. The accuracy of the integrated solution increases as the window length increases. Unfortunately a longer window results in an increase in computation time.

From the results above a window length of 5 secs was chosen to continue this research. A window length of 1 second proved to be too short to produce an accurate solution particularly in the case of the integrated solution (see figures 3.70 and 3.71). A window length of 10 secs produced an integrated solution which closely matched the optimal solution in the nonlinear controller case. The solution produced by a window length of 5 secs, while not as precise, still managed to develop a solution closely resembling the optimal solution. For this reason a window length of 5 seconds is seen as a good compromise between efficiency and accuracy. Hence for the rest of this research a window length of 5 seconds will be used along with cost type 5 and an integration time step of 0.01secs.

The next sub-section will look at the effect the initial condition has on the overall solution.

3.5.4.3 Effect of Initial Condition

The initial condition is another factor which must be considered in the design and selection of the controller. The sensitivity of the starting point on the overall solution is critical particularly in the case on linear techniques. This sub-section investigates the difference between the solutions produced by the linear and nonlinear controllers as a function of the initial condition. Again the path is $y = 5$ and the initial y is varied from 0m to 10m in steps of 0.1m. The errors between the nominal path and the actual robot position are calculated at various points along the prediction horizon namely at 1 sec, 2 secs, 3 secs, 4 secs and 5 secs for all initial y values. Plots of errors versus initial y for the different time are given in figures 3.76 to 3.80. The plots given in these figures show the errors between the optimal solution and the nominal path as well as the errors between the integrated solution and the nominal path for both the linear and nonlinear controllers. The errors arising from the integrated solution of the linear controller are shown on a separate plot underneath the main plots as the errors were much higher compared to the others and by plotting all errors on the one graph the errors produced by the other solutions were not as clearly visible. The results show that as the time increases from 1 second to 5 secs the errors decrease as the robot approaches the nominal path. The results clearly show that the further away the robot is from the nominal path (i.e. the greater the perturbation) the higher the error in the case of the linear controller. At the 1 second mark along the prediction window (figure 3.76) the errors between the solution produced by the nonlinear controller and the nominal path ($y = 5$) are seen to be linear as a function of initial y. Moving further along the prediction window (figures 3.77, 3.78, 3.79 and 3.80) shows that these errors decrease and are very close to zero for any y_0. There is only a small region around the nominal path, $y = 5$, during which the errors produced by the linear controller are zeros and match those produced by the nonlinear solution at any time along the prediction window.

Figure 3.81 shows plots of the errors between the optimal solution and the integrated solutions for both linear and nonlinear controllers. The integrated solution can be seen to be in compliance with the optimal for the nonlinear controller solution however discrepancies are present between the integrated and optimal solutions produced by the linear controller. The results show that there is only a small region around $y = 5$ for which the solution produced by the linear and nonlinear controllers overlap.

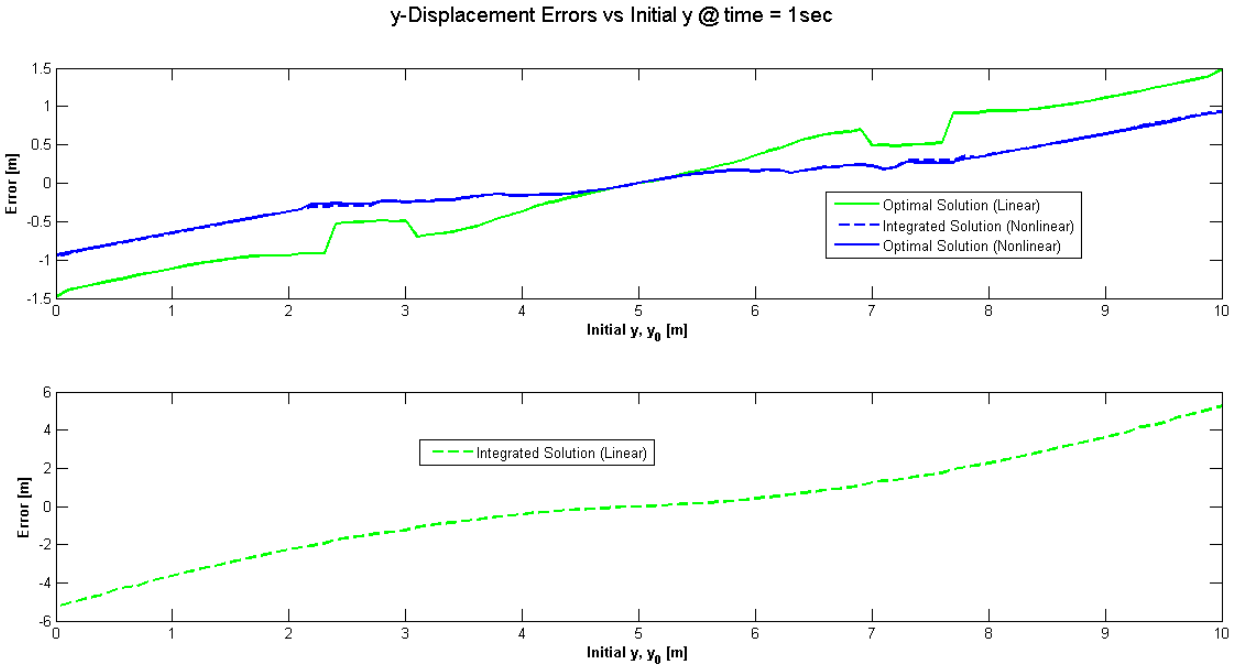

Figure 3.76: Open Loop: Initial Conditions vs y-Displacement Error, time = 1 sec

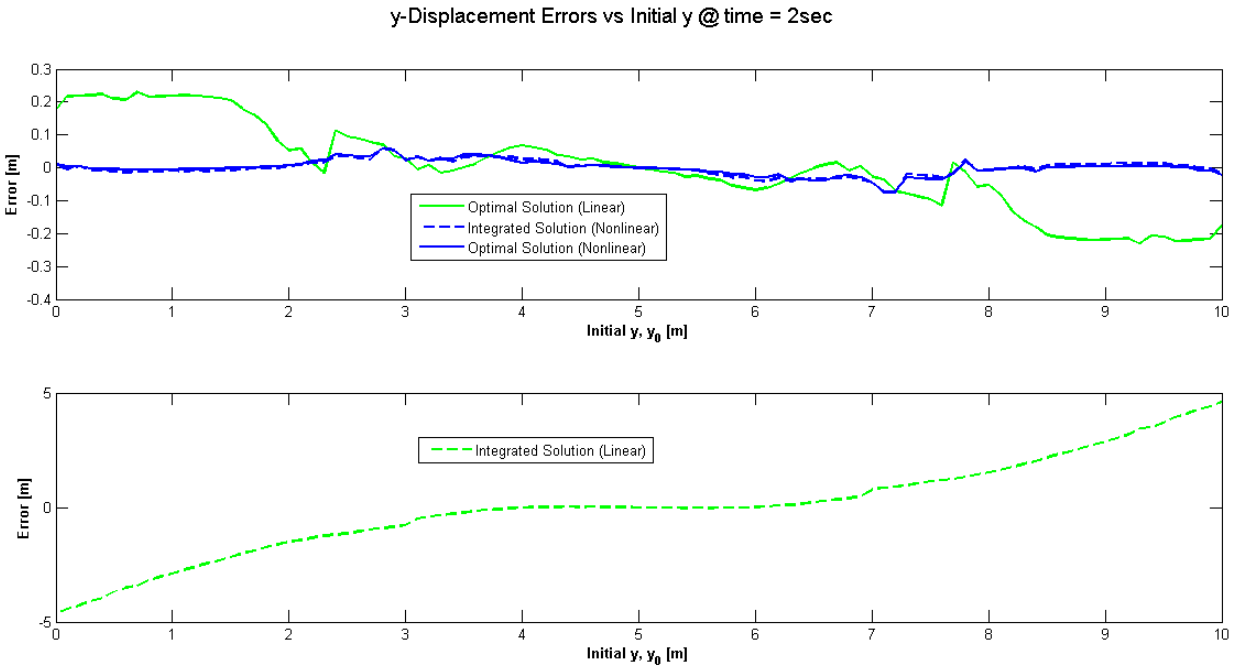

Figure 3.77: Open Loop: Initial Conditions vs y-Displacement Error, time = 2 secs

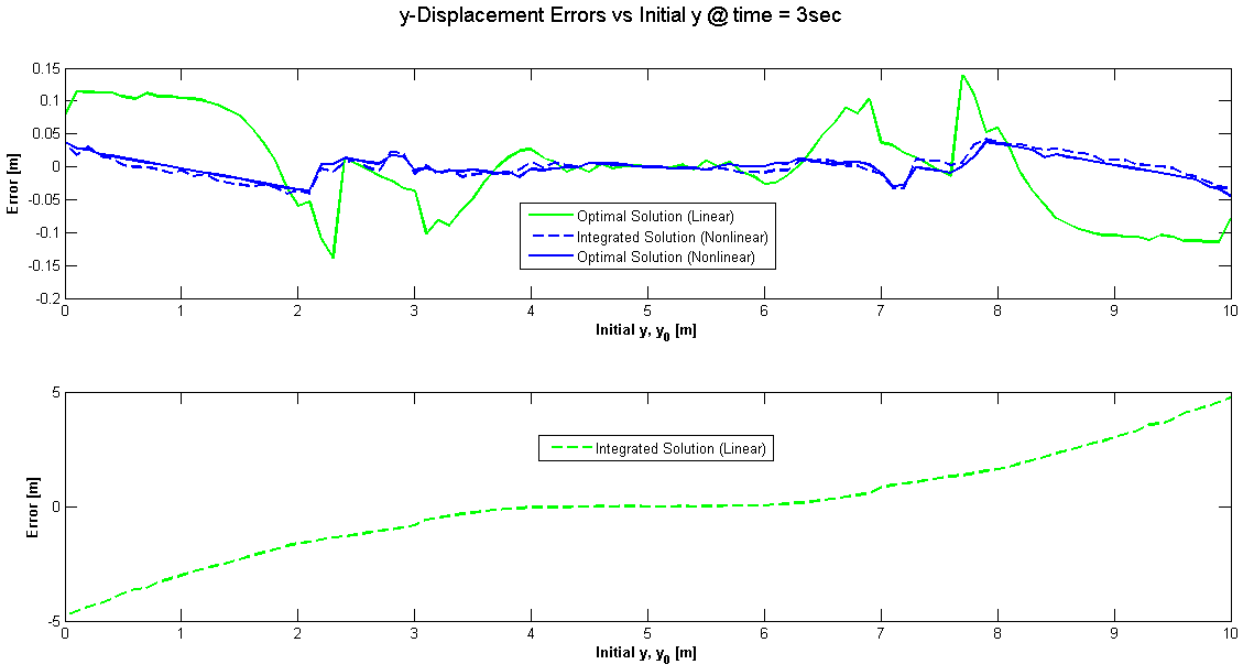

Figure 3.78: Open Loop: Initial Conditions vs y-Displacement Error, time = 3 secs

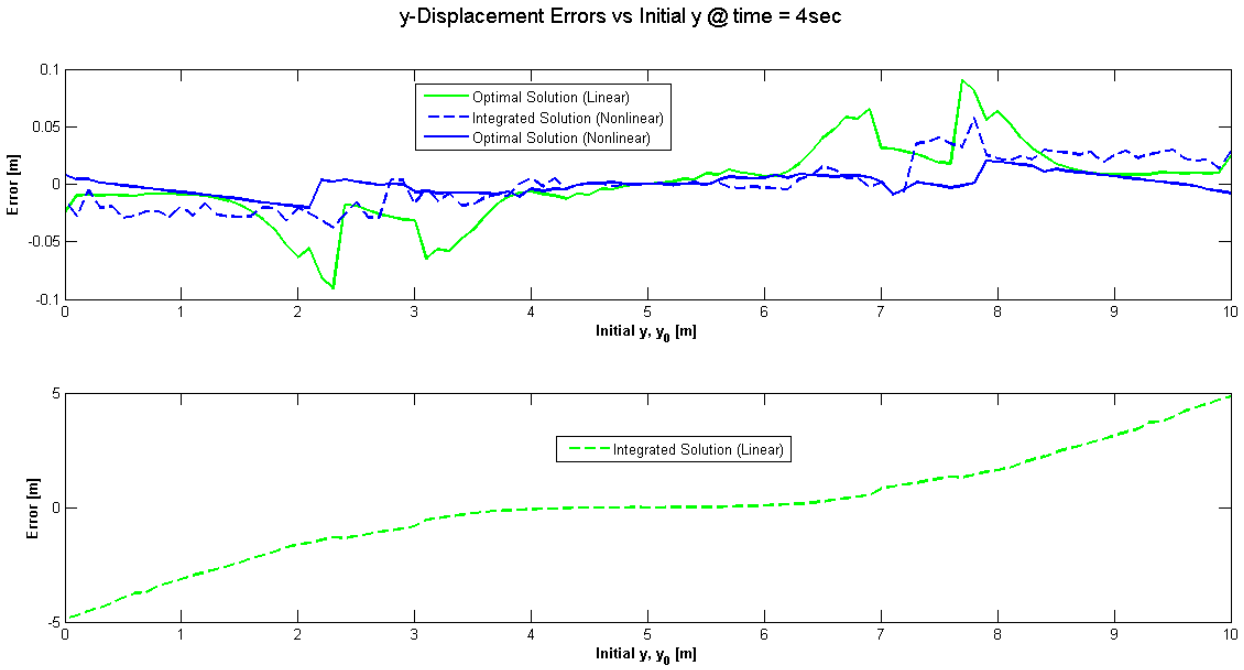

Figure 3.79: Open Loop: Initial Conditions vs y-Displacement Error, time = 4 secs

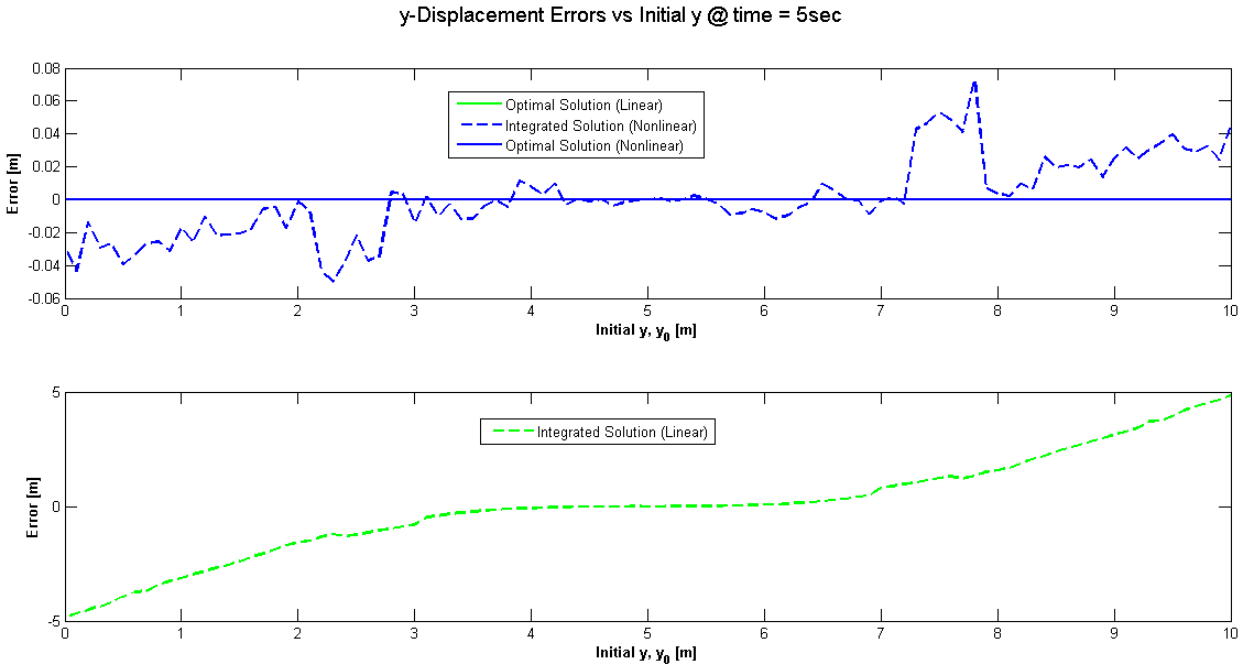

Figure 3.80: Open Loop: Initial Conditions vs y-Displacement Error, time = 5 secs

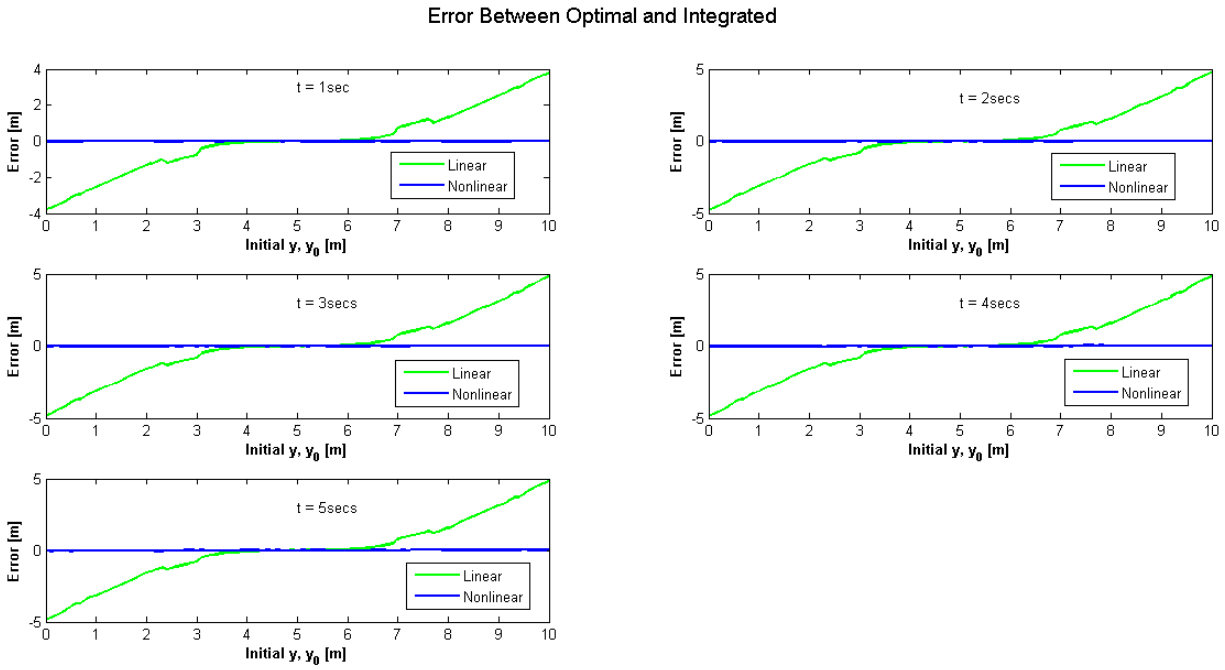

Figure 3.81: Initial Conditions vs y-Displacement Error Between Optimal and Integrated

The next subsection investigates the closed loop problem and compares the solution of both linear and nonlinear MPC.

3.5.5 The Closed Loop Problem

The aim of these simulations is to implement and investigate the behaviour of both linear and nonlinear MPC regarding the closed loop problem. The fault tolerant problem is essentially closed loop hence to apply NMPC to fault tolerant control the pseudosepectral NMPC method is first tested on the simpler 2D robot model.

To test the models the reference trajectory given in figure 3.82 was used.

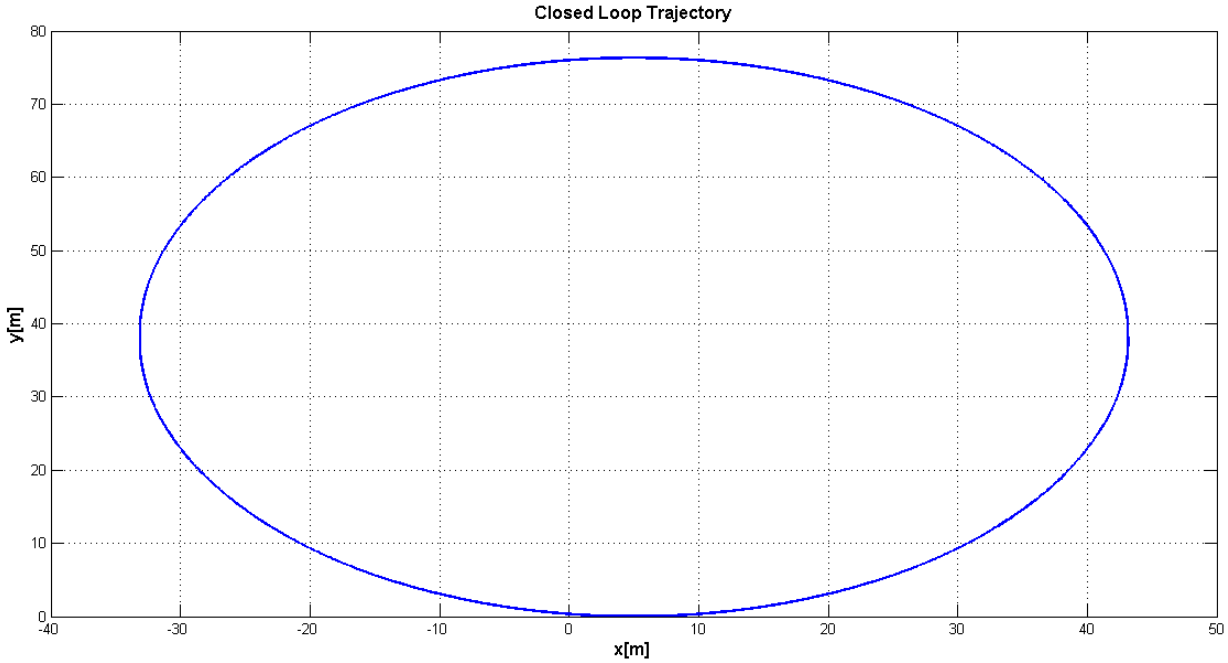

Figure 3.82: Reference Trajectory

Based on the analysis from the previous subsections a prediction window length of 5 seconds was used with 50 collocation points, cost type 5 and and integration time step of 0.01secs. Constraints of $\pm$1000deg/sec are placed on the control inputs which are the angular velocities produced by the right and left wheels. Three different scenarios were set up:

Scenario 1: Robot begins on the path with initial conditions $y_0 = [5, 0, 0]^\intercal$.

Scenario 2: Robot begins slightly off the path with initial conditions $y_0 = [-2, 4, 0]^\intercal$.

Scenario 3: Robot begins well off the path with initial conditions $y_0 = [0, 20, 0]^\intercal$.

For stability $H_u = H_p$ [4]. In many MPC \NMPC formulations H_u is less than H_p. While this greatly reduces the computational expense it does however produce a suboptimal solution [105],

hence for the purposes of this research the control horizon is equal to the prediction horizon.

The trajectory plot for scenario 1 is given figure 3.83. Here the robot begins on the path and both the linear and nonlinear solution can be seen to produce the same solution, something that is also clear from the plots of optimal inputs produced by both controllers as given in figure 3.84. The red lines in figure 3.84 show the constraint boundaries.

The trajectory plots for scenario 2 are presented in figure 3.85. Here the robot begins slightly off the path. Both the linear and nonlinear controllers manage to bring the robot back onto the path. The plots of the optimal inputs (figure 3.86) show that initially both controllers work at the maximum constraint to drive the robot back onto the path. Once the path is reached (i.e. perturbations are small) both controllers exhibit the same performance.

The trajectory plots for scenario 3 presented in figure 3.87 show the robot starting completely off the path (large perturbation). In this case only the nonlinear controller is able to bring the robot back onto the path with the linear controller unable to drive the robot back to the path. Figure 3.88 shows the inputs produced by both controllers with the plots clearly showing that the linear controller works very hard to take the robot back onto the path by consistently working at the constraints however is still unable to return the robot back to the path.

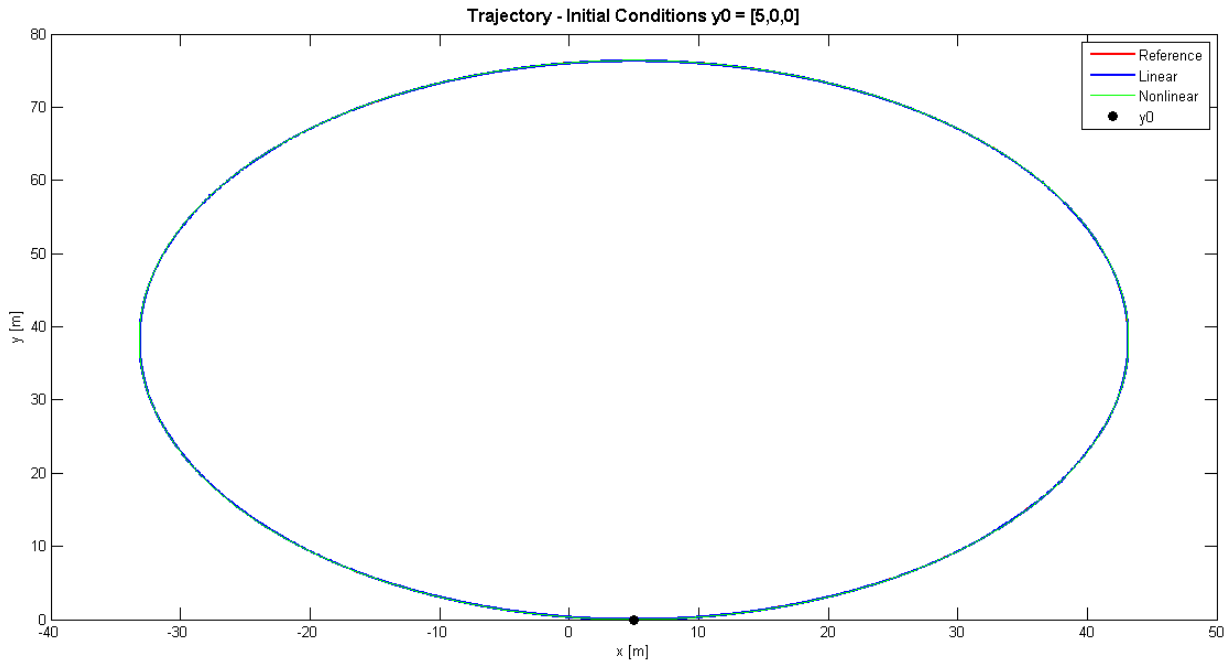

Figure 3.83: Closed Loop: Scenario 1 - Trajectory

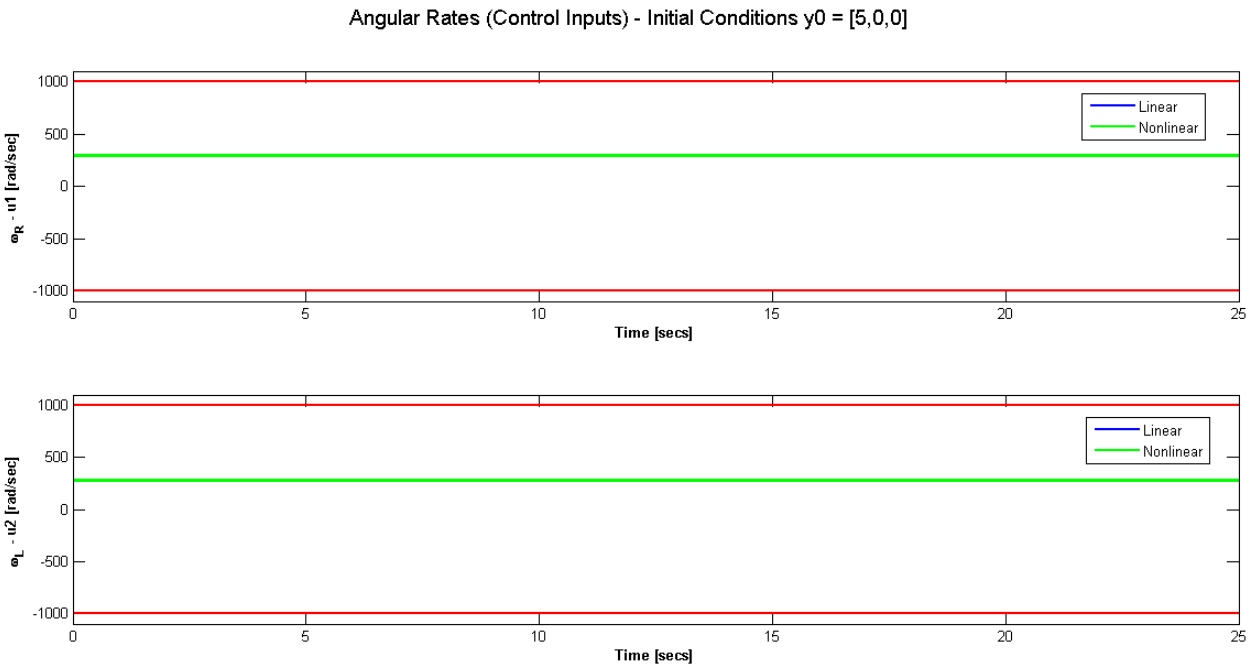

Figure 3.84: Closed Loop: Scenario 1 - Angular Rates

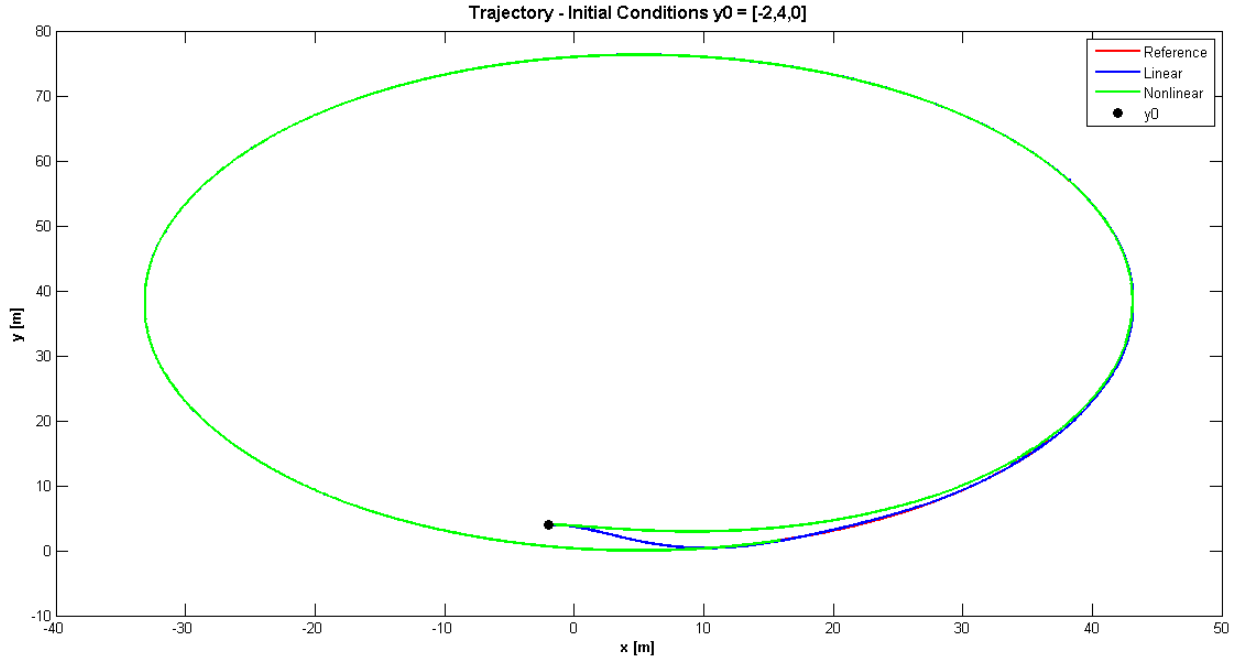

Figure 3.85: Closed Loop: Scenario 2 - Trajectory

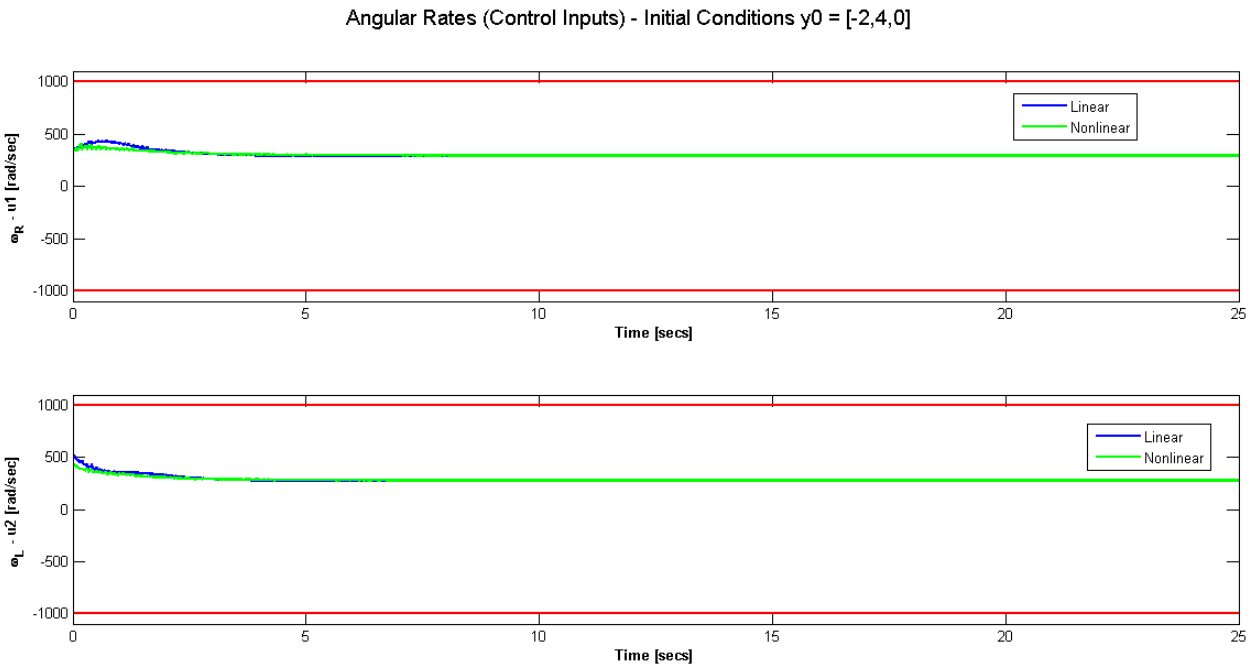

Figure 3.86: Closed Loop: Scenario 2 - Angular Rates

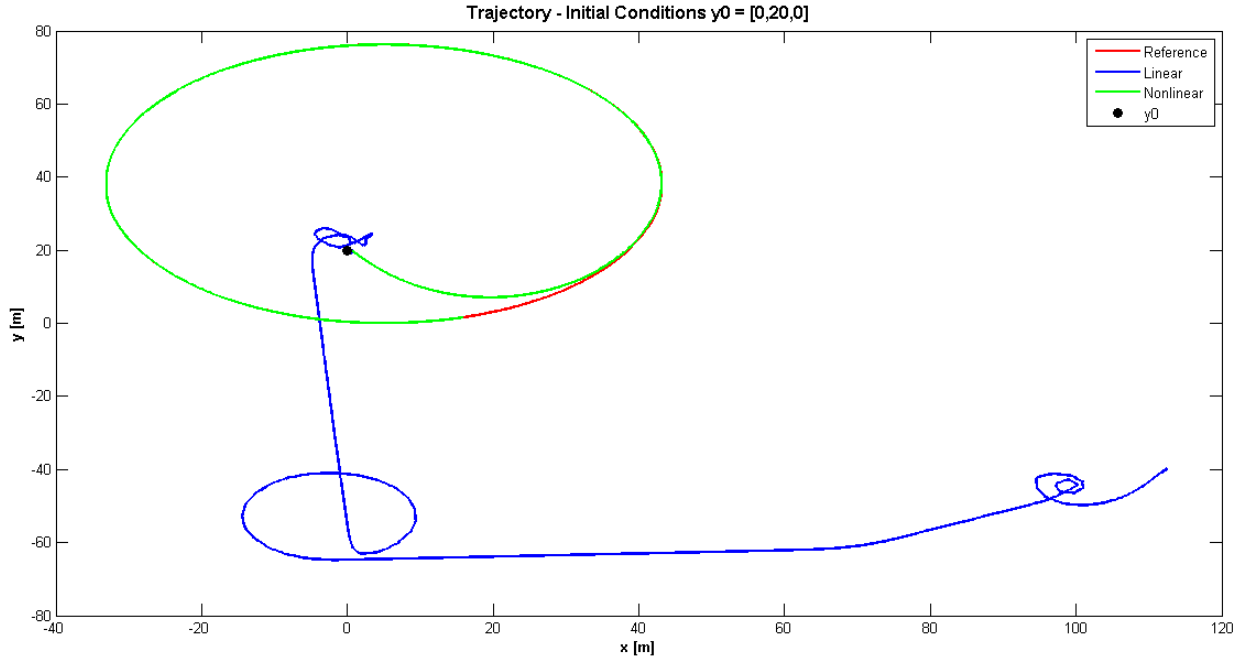

Figure 3.87: Closed Loop: Scenario 3 - Trajectory

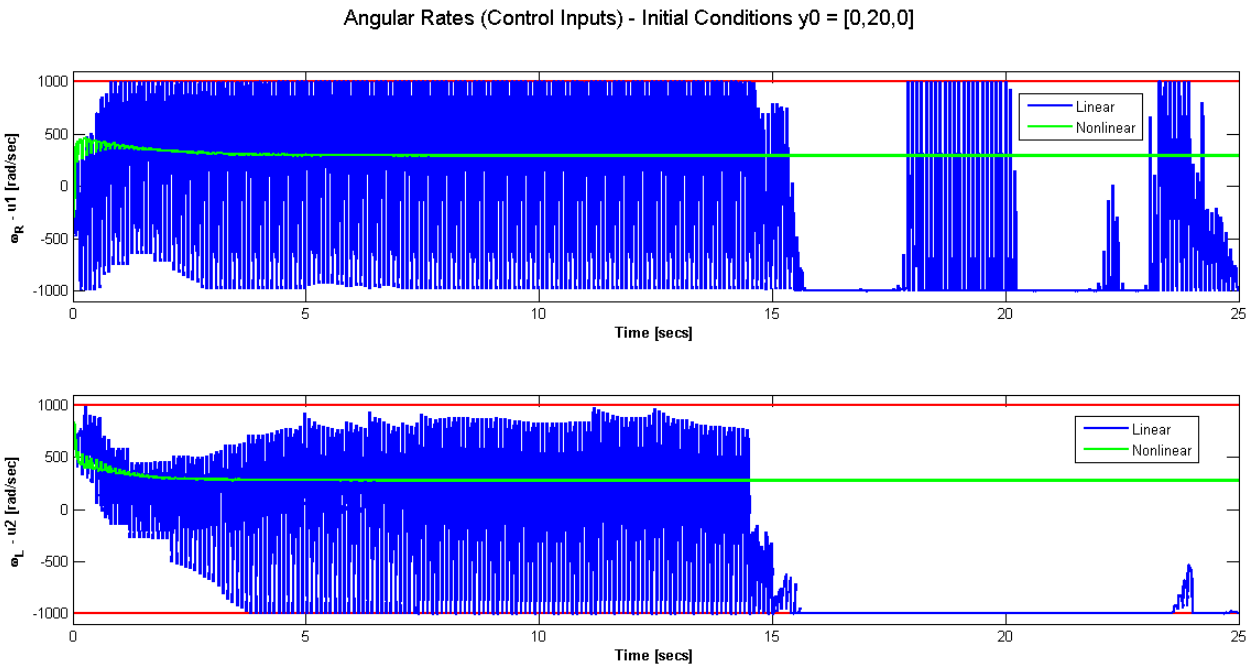

Figure 3.88: Closed Loop: Scenario 3 - Angular Rates

The results clearly show that the pseudospectral NMPC solution to the nonlinear model predictive controller is a viable choice outperforming its linear counterpart when the perturbations are large.

3.6 Summary of Findings

This chapter was dedicated to the theoretical and practical implementation aspects of MPC. A viable solution was sought in particular for nonlinear MPC. The Pseudospectral numerical method was found to be the best candidate. The solution was applied to a 2D robot model in both open and closed loop settings. Comparisons were made between linear MPC and the nonlinear MPC solutions. For small perturbations the two controllers produced the same results. However for large perturbations where nonlinearity effects are more significant the pseudospectral nonlinear MPC controller was found to produce more accurate results than the linear MPC controller.

The next chapter will investigate the application of this solution to fault tolerant control.

Chapter 4

Fault Tolerant Control System Design

4.1 Introduction

4.1.1 Motivation

After examining the various optimal control techniques for the implementation of nonlinear model predictive control (NMPC) in chapter 3, it became clear that pseudospectral discretisation provided the most viable solution. In order to develop this control technique into a full-fledged fault tolerant control system, it is imperative for fault detection and identification (FDI) to be incorporated into the model. This is the aim of this chapter.

Fault tolerant control systems are classified as either Passive or Active (see sub-sections 2.3.1 and 2.3.2). A passive fault tolerant control (FTC) system "passively" handles faults, which are not explicitly detected and isolated and about which no information is gathered. The controller is, thus, designed to automatically counteract only particular faults. The benefit of a passive controller is that performance is unaffected by fault detection delays or false alarms. However the handling of only faults that are incorporated into its design is a major drawback of a passive fault tolerant control (FTC) system, as all other faults may cause the system to become unstable. Another disadvantage of a passive FTC controller is its reliance on robust control techniques that can be overly conservative. Active FTC systems, on the other hand, "actively" seek the fault and try to gather as much information about it as possible to help the controller overcome any consequential instabilities.

Fault detection and identification (FDI) is a key component in an active fault tolerant control (FTC) system and is the most difficult aspect of FTC [26]. In his 1997 paper Patton [97] states that most research on FDI has been done independent of the controller design and no combined design exists. Since then there has been some research on the integration of FDI and FTC; however much remains to be done in this area.

According to Zhang and Jiang [148], FDI can be either parameter based or state based. State based fault detection systems have shorter time delays but do not provide enough detail for fault diagnosis, leading to parameter based systems being preferred. In general the FDI system has three major tasks:

1. Fault detection indicates that something is wrong in the system i.e. the occurrence of a fault and the time of its occurrence.

2. Fault isolation determines the location and the type of the fault (the manner in which the component has failed).

3. Fault identification determines the magnitude (size) of the fault.

To design an active FTC system, the aim of this thesis, it is essential that the final design includes an FDI subsystem. I investigate existing FDI techniques in this chapter, and use the knowledge to reconfigure the fault tolerant controller that I started designing in Chapter 3.

The simple 2D robot model introduced in section 3.5 is used as an illustrative example.

4.1.2 Outline

Section 4.2 presents a brief literature survey of the techniques used in fault detection and identification as applied to flight control. This is followed by an in-depth investigation of the FDI methods chosen for the work in this thesis, namely the extended Kalman filter (EKF), the unscented Kalman filter (UKF) and the interactive multiple model (IMM), in section 4.3.

An extended Kalman filter (EKF) filter, unscented Kalman filter (UKF) filter, EKF based interactive multiple model (IMM) filter and a UKF based IMM filter are each designed in section 4.4 for the purposes of FDI using the 2D robot model of section 3.5. Hence four different active fault tolerant control systems using NMPC as the controller design are formulated and implemented

in MATLAB for the 2D robot model.

Each of the four different active FTC systems developed in section 4.4 are tested under different conditions in section 4.5. The tests are designed to evaluate the performance of the filters as well as the interaction of the filter and controller designs. The section provides a detailed analysis of the test results and concludes with a summary of findings. Based on the findings the filter chosen to continue the research is implemented for active FTC using linear model predictive control (MPC) as the controller and the results are compared to its nonlinear counterpart.

Finally a brief conclusion is given in section 4.6.

4.2 Literature Review of FDI

Fault Detection and Identification is a very mature field of study and provides many powerful quantitative and/or qualitative modelling tools as well as artificial intelligence. As Patton [97] identified back in 1997, most FDI research does not include a combined controller and FDI design. The main difficulty with active FTC is on-line reconfiguration which requires detailed information about changes in system parameters. Hence the main role of the FDI subsystem, particularly in this research, is the gathering of information on parameter changes to assist in controller reconfiguration.

Most reconfigurable controllers use real time estimates of the system parameters provided by parameter estimation based FDI. These approaches to FDI are thought to provide the controller with system information in a more suitable format for on-line reconfiguration than the alternative approaches based on state estimation. Many difficulties still exist in parameter based estimation techniques. For example, in order to get good estimates it may be necessary to introduce perturbation signals to ensure all plant modes have been sufficiently excited; this is particularly true for aircraft.

Observer based FDI schemes are very dependent on the models upon which the scheme is designed. Hence the reliability of these FDI systems must be higher than the monitored system. The better the model representing the dynamic behaviour of the system, the higher the chance of improving the reliability and performance in detecting and isolating faults. Plant model mismatches can cause false alarms, or even miss faults, and hence robustness issues in FDI are very

important [8]. One of the advantages of MPC is the easy incorporation of robust control ideas [26].

As mentioned in chapter 2, hardware redundancy and full self diagnosis equipment are not always feasible, especially for unmanned aerial vehicles (UAVs), because of their cost and associated weight penalties. The FTC system that I wish to design is based around analytical redundancy (AR), which was discussed in sub-section 2.1.3, an approach that utilises the available data and mathematical model of the plant (model-based FDI) for fault tolerance. This approach is based on the belief that a fault changes physical parameters and hence the dynamical model of the plant.

The next section gives a brief overview of the current literature on the application of fault detection and identification to flight control.

4.2.1 Application to Flight Control

Fault tolerance has been applied to many industries. The focus of this research is flight control, and this section looks at the FDI schemes applied specifically to flight vehicles, in particular UAVs.

Fault detection can be divided into three categories [96]:

1. Knowledge based - these techniques use artificial intelligence methods such as neural networks (NNs) or fuzzy decision logic to detect and classify faults.

2. Signal processing - these techniques use signal characteristics such as spectrum information or statistical information to enable the generation of signals that can be used to provide an indication of the existence of a failure.

3. Model based - these techniques are similar to signal processing techniques except that a model is utilised to estimate/measure the values provided by error signals (residuals) to provide an indication of the existence of a failure.

A report by Cork et. al [33], presents an NN based sensor fault detection scheme to provide analytical redundancy (AR) from sensors already existing onboard a UAV. Artificial intelligence schemes for flight control FDI are described in Lin and Liu [81], with the FDI based on detecting immediate changes in the correlation between pitching, yawing and rolling moments. An NN

is used for FDI and adaptive compensation, while failure tolerance is addressed by applying a wavelet neural net-based proportional integral derivative (PID) control law. Perhinschi et. al [100] present an FDI scheme for flight control capable of detecting and identifying failures affecting aircraft actuators, sensors, structural integrity and propulsion, using ideas from the modelling of bioimmune systems, combined with other artificial intelligence (AI) techniques. An immunity based fault detection system operates in a similar manner to the human immune system whereby a distinction is made between the entities that belong to the organism and those that do not.

Observer based FDI is the most common approach studied in recent literature. An application to flight control can be found in Wang et. al [121] where the focus is on control surface failures. Here the FTC system encompasses two controllers; the main controller designed for the faultless system with the second controller being a compensator designed to handle the faulty case. Fault detection is then based on an extended state observer to estimate faults. Liu et. al [82] apply an adaptive fault estimation observer based algorithm for linear discrete systems with actuator faults to a model of the F-16 aircraft.

Boskovic et. al [20] suggest a viable approach to FDI for air and space vehicles through the use of multiple model switching and tuning (MMST). The authors claim that most techniques are based on treating failures and structural damage as parametric uncertainty and then designing corresponding adaptive controllers that achieve the desired control response. The standard approach to indirect adaptive control is based on certainty equivalence, where certain plant parameters are estimated online and the estimates are, in turn, used by the controller to assure stability of the overall system (Boskovic et. al [20]). However such approaches, when applied to a linearised aircraft model, can only handle small to moderate uncertainty and, with regards to FTC, only a small set of failures can be tolerated. Boskovik and co-authors [20] suggest using a MPC multiple model based method, where different models are used to describe the dynamics of the system for different operating regimes. These are referred to as identification models as they identify the current dynamics of the system.

Sliding mode observers for fault detection have been used by Edwards et. al [41]. Historically, sliding mode methods have generated interest because of their strong robustness to a particular class of uncertainty. This is achieved by using nonlinear control/injection signals to force the

system trajectories to attain motion along a surface in the state-space in finite time. Sliding mode observers are able to reconstruct unmeasurable signals within a process by appropriate scaling and filtering of the so-called "equivalent output error injection". The fault tolerant controller in Edwards et. al [41] is also based on a state feedback sliding mode scheme and was tested on the GARTEUR flight simulator (Edwards et. al [41]).

Yee et. al [141] illustrate an FDI scheme based on estimating the fault through matrix algebra. Simulations are based on a combat aircraft and the scheme shows that it has the potential to handle multiple faults. Meanwhile Xiao-song [132] claims that FDI cannot be performed by a single method and combines statistical and analytical redundancy approaches. A set of robust adaptive observers are set up for residual generation (analytical) alongside a statistical method of residual evaluation (statistical) to detect faults online (Xiao-song [132]), and the method is applied to a model of a fighter aircraft.

Ducard's book [39] details a fault tolerant control scheme for a small UAV. The FDI presented therein is based on a particular type of multiple model adaptive estimation (MMAE) called the extended multiple model adaptive estimation (EMMAE). A bank of extended Kalman filters are designed to monitor the health of each actuator, and the method allows estimation of the fault control signal by the respective EKF. By allowing actuator deflection estimates to be a part of the system state vector, the EMMAE method is able to work for all possible positions where an actuator can be locked or floating. Ducard [39] discovered that failures near trim conditions are harder to detect and isolate, so a control allocation scheme is used where estimates from the FDI are used to solve a set of algebraic equations that, in turn, are used to describe the dimensionless aerodynamic coefficients C_L, C_M and C_N. Given five actuators the system is faced with three equations and five unknowns, and to obtain a unique solution various behaviour modes are set up. Ducard [39] states that this control allocation method is simple, fast and can be implemented on a small processor or a microcontroller where computational power is limited. The final overall solution is a reconfigurable guidance system where a new flight path is calculated based on information about the fault.

The next section describes in detail the particular FDI techniques chosen for this research. The selected methods are the EKF, the UKF and the IMM filters.

4.3 Fault Detection Techniques Selected for Implementation: Theoretical Description

The fault detection techniques considered here are all based on filtering techniques, namely the EKF, the UKF, the EKF based IMM and the UKF based IMM. These filters are used to sequentially estimate the state of a dynamic system using a sequence of noisy measurements made on the system. The state estimates are then utilised to aid in fault detection and control reconfiguration. A general overview and key mathematical concepts are provided for each method.

4.3.1 Extended Kalman Filter (EKF)

The EKF is an extension of the well known Kalman filter. One of the drawbacks of the Kalman filter is that it does not provide good estimations for nonlinear systems [106]. The EKF approximates (or linearises) the nonlinear functions in the state dynamic and measurement models. There are two main stages during an EKF (and the general Kalman filter) cycle: predict and update. During the prediction stage the filter states and covariances are predicted forward one time step as are the measurement predictions. During the update stage corrections are made to the state predictions via noisy measurements.

The extended Kalman filter is derived for nonlinear systems with additive noise. A summary of the EKF (Ristic et. al [106]) equations are given below. The target state $\mathbf{x}_k$ and measurement $\mathbf{z}_k$ equations propagate according to:

$$\mathbf{x}_k = \mathbf{f}_{k-1}\left(\mathbf{x}_{k-1}\right) + \mathbf{v}_{k-1}, \tag{4.1}$$

$$\mathbf{z}_k = \mathbf{h}_k\left(\mathbf{x}_k\right) + \mathbf{w}_k, \tag{4.2}$$

where $\mathbf{v}_{k-1}$ and $\mathbf{w}_k$ are random sequences and are mutually independent with zero-mean, white Gaussian with covariances $\mathbf{Q}_{k-1}$ and $\mathbf{R}_k$ respectively. The EKF is based on the assumption that local linearisation of the above equations may be a sufficient description of nonlinearity. The mean and covariance of the underlying Gaussian density are computed recursively in a two stage process (Ristic et. al [106]):

Stage 1: Prediction

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}\left(\hat{x}_{k-1|k-1}\right), \tag{4.3}$$

$$\mathbf{P}_{k|k-1} = \mathbf{Q}_{k-1} + \hat{\mathbf{F}}_{k-1}\hat{\mathbf{P}}_{k-1|k-1}\hat{\mathbf{F}}_{k-1}^{\top}. \tag{4.4}$$

$$\tag{4.5}$$

Stage 2: Update/Correction

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\nu_k, \tag{4.6}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^{\top}, \tag{4.7}$$

where

$$\nu_k = \mathbf{z}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1}\right), \tag{4.8}$$

$$\mathbf{S} = \hat{\mathbf{H}}_k\mathbf{P}_{k|k-1}\hat{\mathbf{H}}_k^{\top} + \mathbf{R}_k, \tag{4.9}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\hat{\mathbf{H}}_k^{\top}\mathbf{S}_k^{-1}. \tag{4.10}$$

The parameter $\mathbf{K}_k$ is commonly known as the Kalman gain and $\mathbf{S_k}$ is referred to as the innovation covariance. The innovation ν_k is the error between the predicted measurement and the actual measurement of the system. The matrices $\hat{\mathbf{F}}_{k-1}$ and $\hat{\mathbf{H}}_k$ are the local linearisation of the nonlinear functions $\mathbf{f}_{k-1}$ and $\mathbf{h}_k$. The two matrices are defined as Jacobians evaluated at $\hat{\mathbf{x}}_{k-1|k-1}$ and $\hat{\mathbf{x}}_{k|k-1}$ respectively (Ristic et. al [106]):

$$\hat{\mathbf{F}}_{k-1} = \left[\nabla_{\mathbf{x}_{k-1}}\mathbf{f}_{k-1}^{\top}\left(\mathbf{x}_{k-1}\right)\right]^{\top}\Big|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1|k-1}}, \tag{4.11}$$

$$\hat{\mathbf{H}}_k = \left[\nabla_{\mathbf{x}_k}\mathbf{h}_k^{\top}\left(\mathbf{x}_k\right)\right]^{\top}\Big|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k-1}}, \tag{4.12}$$

where:

$$\nabla_{\mathbf{x}_k} = \left[\frac{\partial}{\partial\mathbf{x}_k[1]}\cdots\frac{\partial}{\partial\mathbf{x}_k[n_x]}\right]^{\top}. \tag{4.13}$$

Since the Jacobians must be calculated analytically the EKF is often referred to as an analytic approximation. If the functions f_k or h_k are discontinuous, analytical methods cannot be applied. A drawback of the EKF is that it always approximates $p\left(\mathbf{x_k}|\mathbf{Z}_k\right)$ to be Gaussian [106].

If the model is highly nonlinear the non-Gaussianity of the true posterior density will be more evident, for example, it could be bimodal or heavily skewed. In this event the performance of the EKF will significantly degrade.

4.3.2 The Unscented Kalman Filter (UKF)

The unscented Kalman filter (UKF) addresses the issue of non-Gaussianity. The UKF is a part of a family of nonlinear filters, referred to as linear regression Kalman Filters, that are based on statistical linearisation rather than analytical linearisation. The key concept behind these filters is to perform nonlinear filtering using a Gaussian representation of the posterior $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$ through a set of deterministically chosen sample points. The true mean and covariance of the Gaussian density are completely captured by these sample points up to the second order of nonlinearity, with errors introduced in the third and higher order when propagated through a nonlinear transform. The EKF on the other hand is only of first order with errors introduced in the second and higher orders. The filters belonging to this family differ only by the method used to select the sample points i.e. their number, weights and values in the filtering equations are identical and are given below. The UKF uses an unscented transform for the selection of points in an EKF framework (Ristic et. al [106]).

We assume that at time $k-1$ the posterior is Gaussian: $p\left(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}\right) \approx \mathcal{N}\left(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}\right)$. The very first step is to represent this density via a set of N sample points $\mathcal{X}^i_{k-1}$ and their weights W^i_{k-1}, $i = 0, \ldots, N-1$. The prediction step is as follows:

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=1}^{N-1} W^i_{k-1} \mathbf{f}_{k-1}\left(\mathcal{X}^i_{k-1}\right), \tag{4.14}$$

$$\mathbf{P}_{k|k-1} = \mathbf{Q}_{k-1} + \sum_{i=0}^{N-1} W^i_{k-1} \left[\mathbf{f}_{k-1}\left(\mathcal{X}^i_{k-1}\right) - \hat{\mathbf{x}}_{k|k-1}\right] \left[\mathbf{f}_{k-1}\left(\mathcal{X}^i_{k-1}\right) - \hat{\mathbf{x}}_{k|k-1}\right]^\top. \tag{4.15}$$

A set of N sample points:

$$\mathcal{X}^i_{k|k-1} = \mathbf{f}_{k-1}\left(\mathcal{X}^i_{k-1}\right), \tag{4.16}$$

are used to represent the predicted density: $p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right) \approx \mathcal{N}\left(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}\right)$ and the predicted measurement becomes:

$$\hat{\mathbf{z}}_{k|k-1} = \sum_{i=0}^{N-1} W_{k-1}^{i} \mathbf{h}\left(\mathcal{X}_{k-1}^{i}\right). \tag{4.17}$$

The update step is defined as:

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \nu_k, \tag{4.18}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^{\intercal}, \tag{4.19}$$

where

$$\mathbf{K}_k = \mathbf{P}_{xz} \mathbf{S}_k^{-1}, \tag{4.20}$$

$$\mathbf{S}_k = \mathbf{R}_k + \mathbf{P}_{zz}, \tag{4.21}$$

$$\mathbf{P}_{xz} = \sum_{i=0}^{N-1} W_{k-1}^{i} \left(\mathcal{X}_{k|k-1}^{i} - \hat{\mathbf{x}}_{k|k-1}\right) \left(\mathbf{h}_k(\mathcal{X}_{k|k-1}^{i}) - \hat{\mathbf{z}}_{k|k-1}\right)^{\intercal}, \tag{4.22}$$

$$\mathbf{P}_{zz} = \sum_{i=0}^{N-1} W_{k-1}^{i} \left(\mathbf{h}_k(\mathcal{X}_{k|k-1}^{i}) - \hat{\mathbf{z}}_{k|k-1}\right) \left(\mathbf{h}_k(\mathcal{X}_{k|k-1}^{i}) - \hat{\mathbf{z}}_{k|k-1}\right)^{\intercal}. \tag{4.23}$$

As can be seen from the above filter equations there is no explicit calculation of Jacobians. Consequently these filters can be utilised even when the nonlinear functions $\mathbf{f}$ and $\mathbf{h}$ have discontinuities.

The UKF uses the unscented transform to select the sample points $\mathcal{X}_{k-1}^{i}$ and weights W_{k-1}^{i} as described in the next subsection.

4.3.2.1 The Unscented Transform

The unscented transform is a means of calculating the statistics of a random variable which undergoes a nonlinear transform (Ristic et. al [106]). For example, given a random variable $\mathbf{a}$ with mean and covariance $\bar{\mathbf{a}}$ and $\mathbf{P}_a$ respectively, on propagation through an arbitrary nonlinear function $\mathbf{g} : \mathbb{R}^{n_a} \rightarrow \mathbb{R}^{n_b}$ it will produce a random variable $\mathbf{b}$:

$$\mathbf{b} = \mathbf{g}(\mathbf{a}).$$

The first two moments of $\mathbf{b}$ can be computed using the unscented transform by first deterministically choosing $2n_a + 1$ weighted samples points $(\mathcal{A}_i, W_i)$ so that they completely capture the true mean $\bar{\mathbf{a}}$ and covariance $\mathbf{P}_a$ of $\mathbf{a}$. The following scheme satisfies this requirement (Ristic et. al [106]):

$$\mathcal{A}_0 = \bar{\mathbf{a}}, \qquad W_0 = \frac{\kappa}{(n_a+\kappa)}, \qquad i = 0, \tag{4.24}$$

$$\mathcal{A}_i = \bar{\mathbf{a}} + \left(\sqrt{(n_a+\kappa)\mathbf{P}_a}\right)_i, \qquad W_i = \frac{\kappa}{2(n_a+\kappa)}, \qquad i = 1,\ldots,n_a, \tag{4.25}$$

$$\mathcal{A}_i = \bar{\mathbf{a}} - \left(\sqrt{(n_a+\kappa)\mathbf{P}_a}\right)_i, \qquad W_i = \frac{\kappa}{2(n_a+\kappa)}, \qquad i = n_1+1,\ldots,2n_a, \tag{4.26}$$

where κ is a scaling factor such that $\kappa + n_a \neq 0$ and $\left(\sqrt{(n_a+\kappa)\mathbf{P}_a}\right)_i$ is the ith row of the matrix square root $\mathbf{L}$ of $(n_a+\kappa)\mathbf{P}_a$ such that $(n_a+\kappa)\mathbf{P}_a = \mathbf{L}^\top\mathbf{L}$. The weights are normalised $\left(\sum_{i=0}^{2n_a} W = 1\right)$. Each sample point is then propagated through the nonlinear functions $\mathbf{g}$:

$$\mathcal{B}_i = \mathbf{g}\left(\mathcal{A}_i\right) \quad (i = 0,1,\ldots,2n_a),$$

and the first two moments of $\mathbf{b}$ are calculated via:

$$\bar{\mathbf{b}} = \sum_{i=0}^{2n_a} W_i \mathcal{B}_i, \tag{4.27}$$

$$\mathbf{P}_b = \sum_{i=0}^{2n_a} W_i \left(\mathcal{B}_i - \mathbf{b}\right)\left(\mathcal{B}_i - \mathbf{b}\right)^\intercal. \tag{4.28}$$

These estimates are accurate to the second order (third order if $\mathbf{a}$ is Gaussian) of the Taylor series expansion of $\mathbf{g}(\mathbf{a})$. The distance $\mathcal{A}_i$, $i = 1,\ldots,n_a$ from $\bar{\mathbf{a}}$ increases with dimension n_a but this can be controlled by the choice of κ. It should be noted that if κ is negative $\mathbf{P}_b$ can become non-positive semidefinite.

4.3.3 Interacting Multiple Model (IMM)

The IMM belongs to a class of filters called the Gaussian Sum Filters. The main concept is the approximation of the required posterior density $p(\mathbf{x}_k|\mathbf{Z}_k)$ by a Gaussian mixture (Ristic et. al [106]):

$$p\left(\mathbf{x}_k|\mathbf{Z}_k\right) \approx p_A\left(\mathbf{x}_k|\mathbf{Z}_k\right) = \sum_{i=1}^{q_k} w_k^i \, \mathcal{N}\left(\mathbf{x}_k^i; \hat{\mathbf{x}}_{k|k}^i, \mathbf{P}_{k|k}^i\right), \tag{4.29}$$

where w_k^i are weights that are normalised, $\sum_{i=1}^{q_k} w_k^i = 1$. Gaussian sum filters are ideal when the posterior density is multimodal because for multimodal densities there is a performance degradation in both the EKF and UKF. The Gaussian Sum filter chosen for further investigation is the IMM. At time k the state estimate is calculated for each possible current model using r

filters, with each filter using a different combination of the previous model-conditioned estimates called mixed initial condition. The algorithm as outlined in Bar-Shalom et. al [10] is:

Step 1: Calculation of the mixing probabilities. The probability that mode M_i was in effect at time $k-1$ given that M_j is in effect at k conditioned on Z^{k-1} is given by:

$$\mu_{i,j}\left(k-1|k-1\right) \triangleq P\left\{M_i\left(k-1\right)|M_j\left(k\right), Z^{k-1}\right\}, \tag{4.30}$$

$$= \frac{1}{\bar{c}_j} P\left\{M_j\left(k\right)|M_i\left(k-1\right), Z^{k-1}\right\} P\left\{M_i\left(k-1\right)|Z^{k-1}\right\}. \tag{4.31}$$

The above can be written as:

$$\mu_{i,j}\left(k-1|k-1\right) = \frac{1}{\bar{c}_j} p_{ij}\,\mu_i\left(k-1\right), \qquad i,j=1,\ldots,r, \tag{4.32}$$

where the normalising constants are:

$$\bar{c}_j = \sum_{i=1}^{r} p_{ij}\,\mu_i\left(k-1\right), \qquad i,j=1,\ldots,r. \tag{4.33}$$

Step 2: Mixing. The mixed initial condition for the filter matched to $M_j\left(k\right)$ is calculated starting with $\hat{x}^i\left(k-1|k-1\right)$:

$$\hat{x}^{0j}\left(k-1|k-1\right) = \sum_{i=1}^{r} \hat{x}^i\left(k-1|k-1\right)\mu_{i|j}\left(k-1|k-1\right), \qquad i,j=1,\ldots,r, \tag{4.34}$$

and the corresponding covariance is given by:

$$\begin{aligned} P^{0j}\left(k-1|k-1\right) = \sum_{i=1}^{r} \mu_{i|j}\left(k-1|k-1\right) \Big\{ & P^i\left(k-1|k-1\right) \\ & + \left[\hat{x}^i\left(k-1|k-1\right) - \hat{x}^{0j}\left(k-1|k-1\right)\right] \\ & \cdot \left[\hat{x}^i\left(k-1|k-1\right) - \hat{x}^0\left(k-1|k-1\right)\right]^{\intercal} \Big\}, \qquad i,j=1,\ldots,r. \end{aligned} \tag{4.35}$$

Step 3: Mode-Matched Filtering. The estimates of the states and covariances calculated in step 2 above are used as inputs to the filter matched to $M_j\left(k\right)$ which uses $z\left(k\right)$ to determine $\hat{x}^j\left(k|k\right)$ and $P^j\left(k|k\right)$. The likelihood functions associated to the r filters:

$$\Lambda_j\left(k\right) = p\left[z\left(k\right)|M_j\left(k\right), Z^{k-1}\right], \tag{4.36}$$

are calculated using the mixed initial condition and covariance from step 2:

$$\Lambda_j\left(k\right) = p\left[z\left(k\right)|M_j\left(k\right), \hat{x}^{0j}\left(k-1|k-1\right), P^{0j}\left(k-1|k-1\right)\right], \qquad j=1,\ldots,r, \tag{4.37}$$

that is:

$$\Lambda_j(k) = \mathcal{N}\left[z(k); \hat{z}^j\left[k|k-1; \hat{x}^{0j}(k-1|k-1)\right], S^j\left[k; P^{0j}(k-1|k-1)\right]\right], \qquad j = 1, \ldots, r. \tag{4.38}$$

Step 4: Mode Probability Update. The mode probabilities are then updated via:

$$\mu_j(k) = \frac{1}{c}\Lambda_j(k)\,\bar{c}_j, \tag{4.39}$$

where c is the normalisation constant and is given by:

$$c = \sum_{j=1}^{r} \Lambda_j(k)\,\bar{c}_j. \tag{4.40}$$

Step 5: Estimations and Covariance Combination. The output is obtained by combining the model-conditioned estimates and covariances:

$$\hat{x}(k|k) = \sum_{j=1}^{r} \hat{x}^j(k|k)\,\mu_j(k), \tag{4.41}$$

$$\hat{P}(k|k) = \sum_{j=1}^{r} \mu_j(k)\left\{P^j(k|k) + \left[\hat{x}^j(k|k) - \hat{x}(k|k)\right]\left[\hat{x}^j(k|k) - \hat{x}(k|k)\right]^{\mathsf{T}}\right\}. \tag{4.42}$$

Figure 4.1 illustrates the flow of the IMM algorithm given above and has been taken from Bar-Shalom et. al [10].

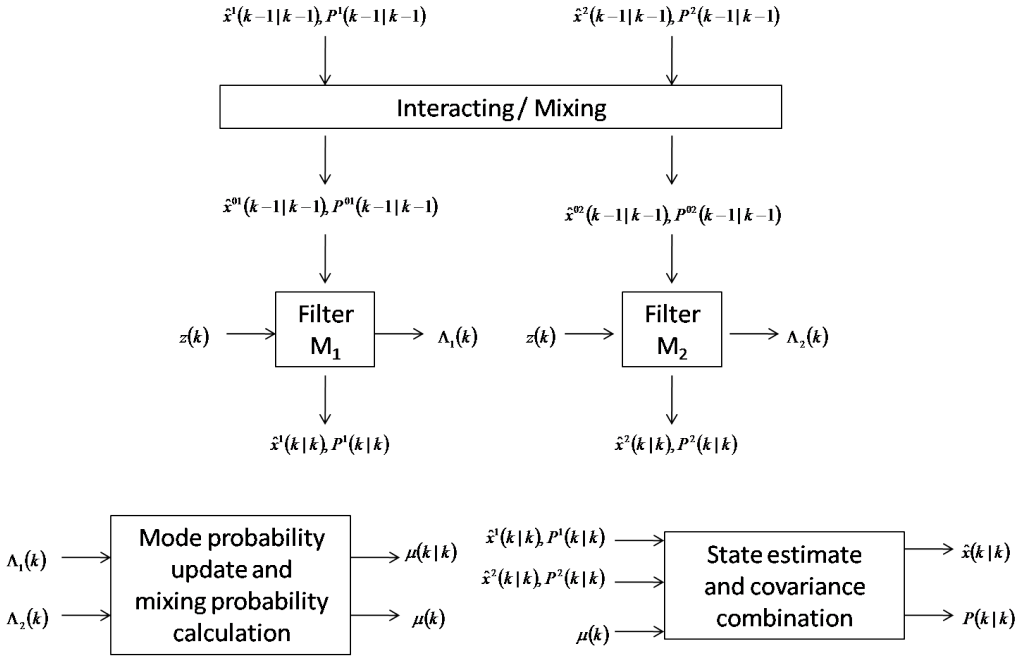

Figure 4.1: IMM Algorithm Flow Chart [10]

This section provided a brief overview of FDI techniques as well as describing in detail the methods chosen for further investigation, the EKF, the UKF and the IMM filters. These techniques are applied to the 2D robot model in the next section.

4.4 Problem Formulation

To test the different filtering techniques in an FDI context the robot model (see figure 3.60 in section 3.5.1) is used again for illustrative purposes.

The fault, which will be simulated and tested for, is a punctured wheel. If a wheel is punctured the radius of the wheel will decrease and so the filters are set up to estimate the radius of the wheel. Four different filters have been designed, the EKF, the UKF, the EKF IMM and the UKF IMM.

The robot parameters used for all simulations are given in table 4.1.

Table 4.1: Simulation parameters for 2D robot model for FDI simulations

Right wheel radius, R_R	$2\,m$
Left wheel radius, R_L	$2\,m$
Distance between wheels, b	$1\,m$
Speed demand	$10\,m/s$
Input constraints on ω_R and ω_L	± 1000 deg / sec

The NMPC controller is constructed using the pseudospectral method with 50 coincidence points and a prediction window length of 5 secs. The filters are updated at 100Hz while the controller is updated at 10Hz.

All work was developed using MATLAB with SNOPT as the nonlinear programming problem (NLP) solver.

4.4.1 EKF Fault Detection Filter

The state vector for the EKF consists of the following states:

$$\mathbf{x} = [x,\ \ y,\ \psi,\ \ R_R,\ \ R_L]^\intercal\,, \tag{4.43}$$

where x, y and ψ are the robot states as defined in subsection 3.5.1 and R_R and R_L are the right wheel and left wheel radii respectively.

The measurements are assumed to be of the speed, V, of the robot:

$$z = V(k) + w(k), \tag{4.44}$$

where $\mathbf{w}(k)$ is additive white noise. The initial state vector and initial covariance matrix are:

$$\mathbf{x}(0) = [x_0,\ \ y_0,\ \psi_0,\ \ 2,\ \ 2]^\intercal\,, \qquad P(0) = \begin{bmatrix} (0.5)^2 & 0 & 0 & 0 & 0 \\ 0 & (0.5)^2 & 0 & 0 & 0 \\ 0 & 0 & (1\pi/180)^2 & 0 & 0 \\ 0 & 0 & 0 & (0.5)^2 & 0 \\ 0 & 0 & 0 & 0 & (0.5)^2 \end{bmatrix}. \tag{4.45}$$

The Q and R noise matrices were chosen to be:

$$Q = \begin{bmatrix} (5\,\Delta t)^2 & 0 & 0 & 0 & 0 \\ 0 & (5\,\Delta t)^2; & 0 & 0 & 0 \\ 0 & 0 & (0.1\,\Delta t)^2 & 0 & 0 \\ 0 & 0 & 0 & (2\,\Delta t)^2 & 0 \\ 0 & 0 & 0 & 0 & (2\,\Delta t)^2 \end{bmatrix}, \qquad R = (0.5)^2, \tag{4.46}$$

where Δt is the update rate of the filter.

For the prediction cycle an Euler integration scheme is used to predict the states of the EKF forward:

$$\dot{\psi} = \frac{R_R}{2b}\omega_R - \frac{R_L}{2b}\omega_L, \tag{4.47}$$

$$\psi(k|k-1) = \psi(k-1) + \dot{\psi}\,\mathrm{dt}, \tag{4.48}$$

$$V(k|k-1) = \frac{R_R}{2}\omega_R + \frac{R_L}{2}\omega_L, \tag{4.49}$$

$$\dot{x} = V\cos(\psi(k|k-1)), \tag{4.50}$$

$$x_k(k|k-1) = x(k-1) + \dot{x}\,\mathrm{dt}, \tag{4.51}$$

$$\dot{y} = V\sin(\psi(k|k-1)), \tag{4.52}$$

$$y(k|k-1) = y(k-1) + \dot{y}\,\mathrm{dt}, \tag{4.53}$$

$$R_R(k|k-1) = R_R(k-1), \tag{4.54}$$

$$R_L(k|k-1) = R_L(k-1). \tag{4.55}$$

The predicted measurement $\hat{z}$ is given by:

$$\hat{z}(k) = V(k|k-1), \tag{4.56}$$

and the F transition matrix (Jacobian matrix) is given by:

$$F = \begin{bmatrix} 1 & 0 & \left(\frac{-R_R}{2}\omega_R\sin(\psi) - \frac{R_L}{2}\omega_L\sin(\psi)\right)\Delta t & \left(\frac{\omega_R}{2}\cos(\psi)\right)\delta t & \left(\frac{\omega_L}{2}\cos(\psi)\right)\mathrm{dt} \\ 0 & 1 & \left(\frac{R_R}{2}\omega_R cos(\psi) + \frac{R_L}{2}\omega_L\cos(\psi)\right)\Delta t & \left(\frac{\omega_R}{2}\sin(\psi(k|k-1))\right)\Delta t & \left(\frac{\omega_L}{2}\sin(\psi)\right)\Delta t \\ 0 & 0 & 1 & \left(\frac{\omega_R}{2b}\right)\Delta t & \left(\frac{-\omega_L}{2b}\right)\Delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.57}$$

N.B. In the matrix above $\psi = \psi(k|k-1)$, $R_R = R_R(k|k-1)$ and $R_L = R_R(k|k-1)$.

Here Δt is the prediction update rate of the filter and has been chosen as $100\,Hz$ or $\Delta t = 0.01\,\text{secs}$. The Jacobian matrix H is given by:

$$H = \begin{bmatrix} 0 & 0 & 0 & \frac{R_R}{2}\omega_R & \frac{R_L}{2}\omega_L \end{bmatrix}. \tag{4.58}$$

Given the above information the Kalman filter equations given in section 4.3.1 are applied to estimate the radius of each wheel in the experiments conducted in section 4.5.

4.4.2 UKF Fault Detection Filter

The general structure of the EKF and UKF are very similar in that they both have a prediction and update cycle and produce a single state vector and a corresponding covariance matrix. For the robot model the state vector is the same as the one given in equation (4.43). The initial state vector, initial covariance matrix, the process noise matrix Q and the noise covariance matrix R all remain the same as those given in subsection 4.4.1. The UKF algorithm given in subsection 4.3.2 is applied to the robot model with $\kappa = 0.001$ [120] (see subsection 4.3.2).

4.4.3 Interacting Multiple Model Fault Detection Filter

The interacting multiple model method, as the name suggests, is made up of multiple models where each model tests a different hypothesis. Four different models (the terms mode and model are used interchangeably and have the same meaning in the context of IMMs) have been designed where:

Model 1: No Fault case.

Model 2: 50% right wheel deflation, left wheel no fault.

Model 3: Right wheel no fault, 50% left wheel deflation.

Model 4: 50% right wheel deflation, 50% left wheel deflation.

During Step 3 of the IMM algorithm given in section 4.3.3 a filter such as the EKF is used to update the states and covariances. This study looks at both an EKF based IMM filter and a UKF based IMM filter.

The initial covariance matrix for each filter and each mode are the same as equation (4.45). The Q and R matrices are those given in equation (4.46) and the initial state vectors for each filter and mode are:

$$\mathbf{x}_1(0) = [x_0,\ y_0\ \psi_0\ 2\ 2]^\intercal\,, \tag{4.59}$$

$$\mathbf{x}_2(0) = [x_0,\ y_0\ \psi_0\ 1\ 2]^\intercal\,, \tag{4.60}$$

$$\mathbf{x}_3(0) = [x_0,\ y_0\ \psi_0\ 2\ 1]^\intercal\,, \tag{4.61}$$

$$\mathbf{x}_4(0) = [x_0,\ y_0\ \psi_0\ 1\ 1]^\intercal\,. \tag{4.62}$$

$$\tag{4.63}$$

The mixing probabilities or mode probabilities are initially set to:

$$\mu = [1/4, 1/4, 1/4, 1/4]^\intercal\,, \tag{4.64}$$

and the mode transition probabilities matrix p is set to:

$$p = \begin{bmatrix} 0.97 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.97 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.97 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.97 \end{bmatrix}. \tag{4.65}$$

4.5 Numerical Results and Analysis

The following reference trajectory is used to test the different fault detection techniques:

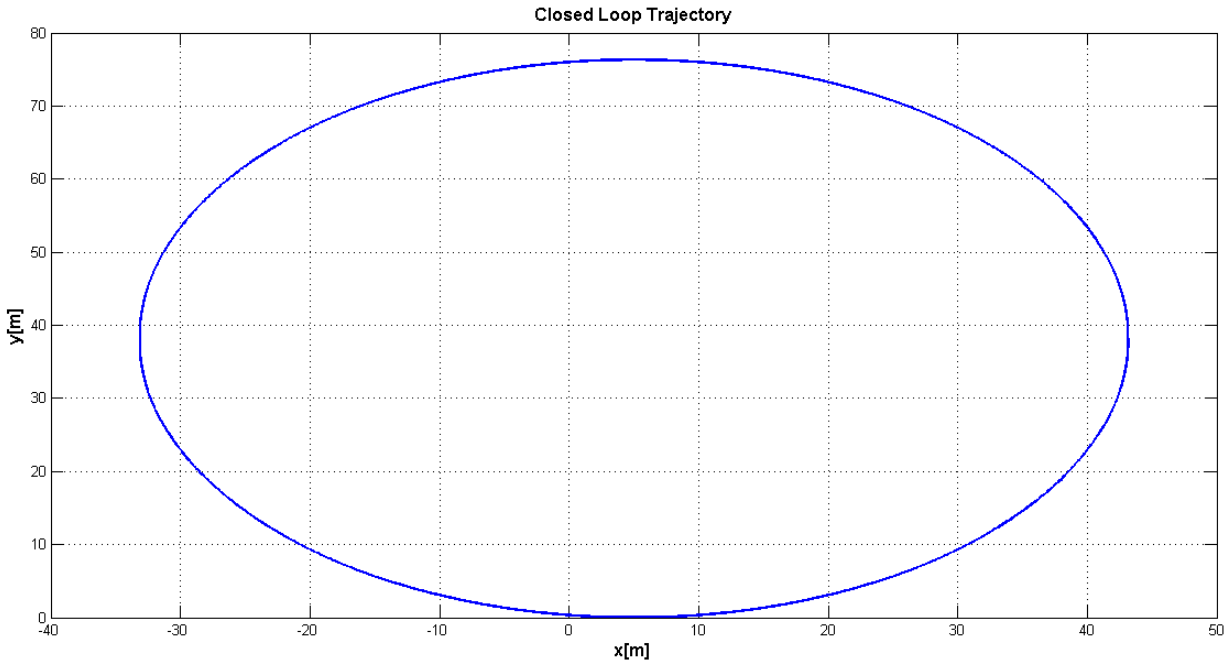

Figure 4.2: FDI: Reference Trajectory

To simulate the measurement additive white noise is added to the speed of the robot which is calculated as a part of the truth simulations of the robot movement.

To test the filters four different test cases were set up and each test case was run twice. During the first run the FDI feedback loop is not closed and the filters are used for estimation only. The FDI loop is closed during the second run to investigate the behaviour of the full active FTC controller. The test cases are as follows:

Scenario 1: No Fault. The objective is to investigate how well the filters estimate the radii of the tyres in a no fault situation.

Scenario 2: Left wheel 50% puncture. In this case a puncture is simulated to occur 10 secs into the simulation. The wheel is assumed to deflate to 50% of its original value instantaneously.

Scenario 3: Left and Right Wheel puncture. In this test case a left wheel puncture is simulated 5 secs into the run and a right wheel puncture is simulated to occur 10 secs into the simulation. Both punctures are assumed to cause an instantaneous reduction of the respective wheel radius to 50% of the original wheel radius.

Scenario 4: Left wheel linear puncture. In this test case once again the left wheel is punctured 10 secs into the run however this time the puncture is assumed to follow a linear reduction in wheel radius according to $R_L = 2 - 0.1t$, where R_L represents the left wheel radius reduced from its original value of 2m down to 0.1m at a rate of 0.1m/s and t is the current time. The radius does not drop to 0m as this caused a complete system failure.

The results for each filter are presented in the next four subsections:

4.5.1 Scenario 1

4.5.1.1 Speed Errors (Innovations)

The plots given in figure 4.3 show the speed innovation (speed errors or correction made as a result of the measurement update, blue solid lines) along with the 2σ uncertainty bounds (dashed red lines) for the EKF. The 2σ uncertainty bounds are a 95% confidence interval and the solution (errors in this case) must remain within the bounds 95% of the time. The figure shows that the EKF filter performs very well for the whole run as the speed errors remain well within the uncertainty bounds throughout the duration of the run both with and without feedback.

The speed innovations produced by the UKF are given in figure 4.4. Similar to the EKF the UKF produces innovations which are well within the 2σ bounds for the whole simulation run.

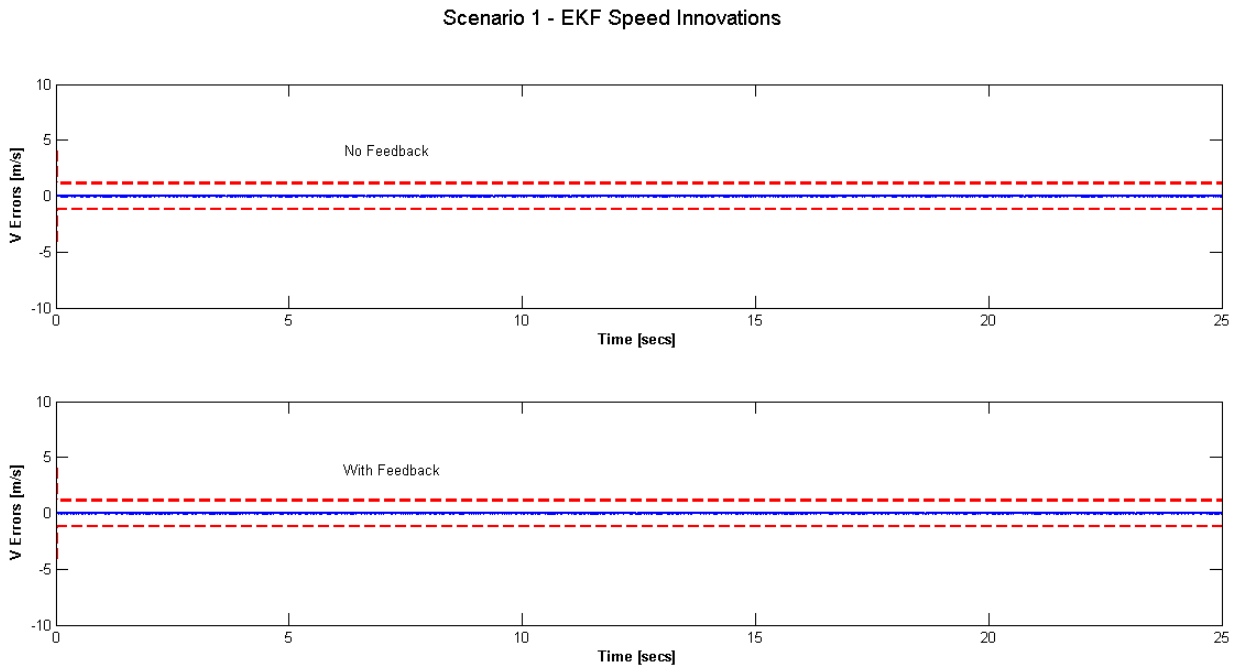

Figure 4.3: Scenario 1 - EKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

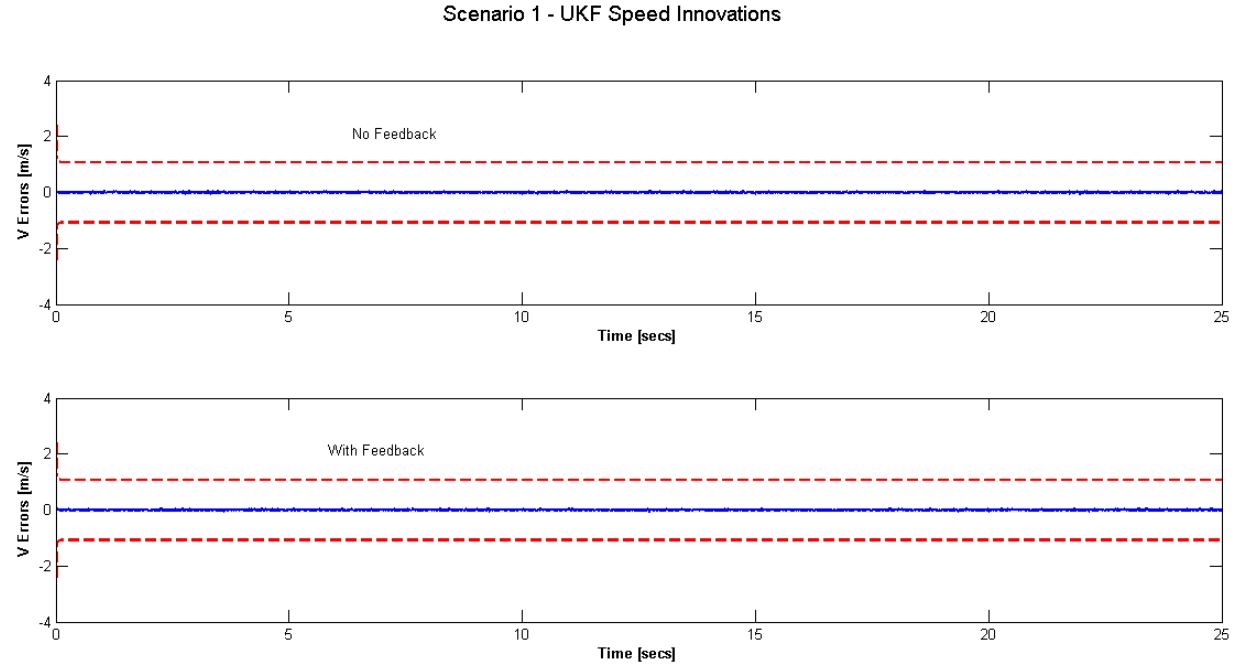

Figure 4.4: Scenario 1 - UKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

The speed error plots for the IMM EKF are given in figure 4.5 for the no feedback case and in

figure 4.6 for with feedback. Both figures show the innovation produced by each model, and the plots show that the filter is able to very quickly detect the model of operation of the robot. In both case, from the value of the uncertainty bounds, the correct model for scenario 1 is model 1 which has the least uncertainty. Compared to the other models, model 1 is the most confident about its estimation of the radius.

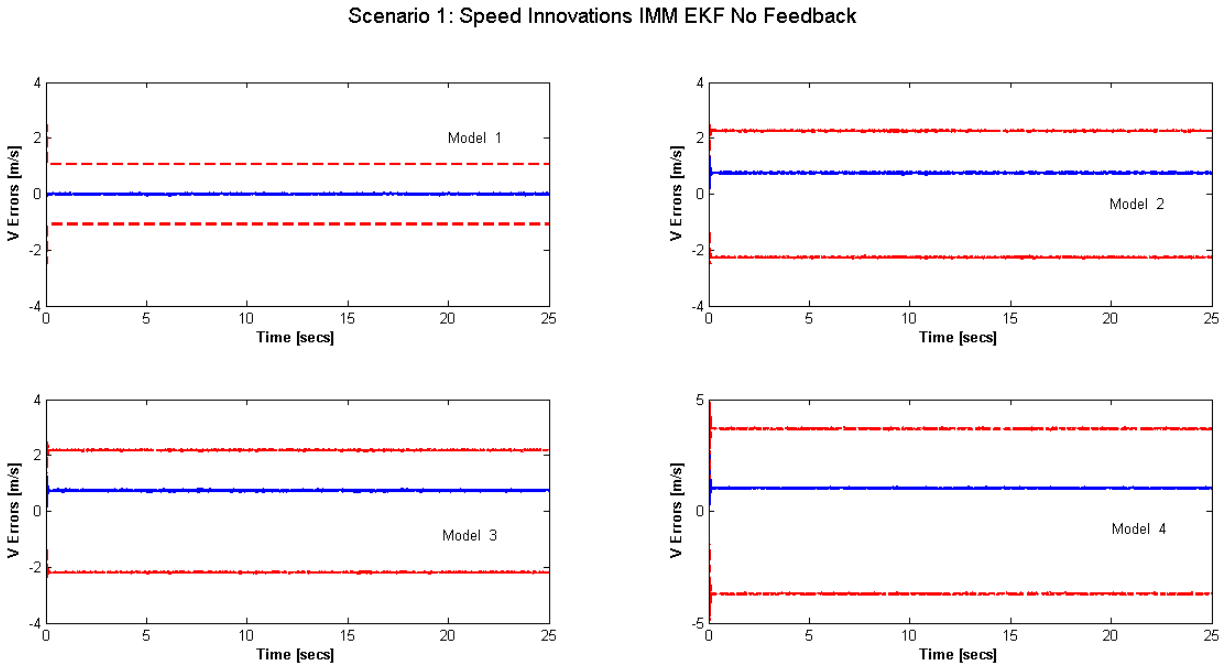

Figure 4.5: Scenario 1 - IMM EKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

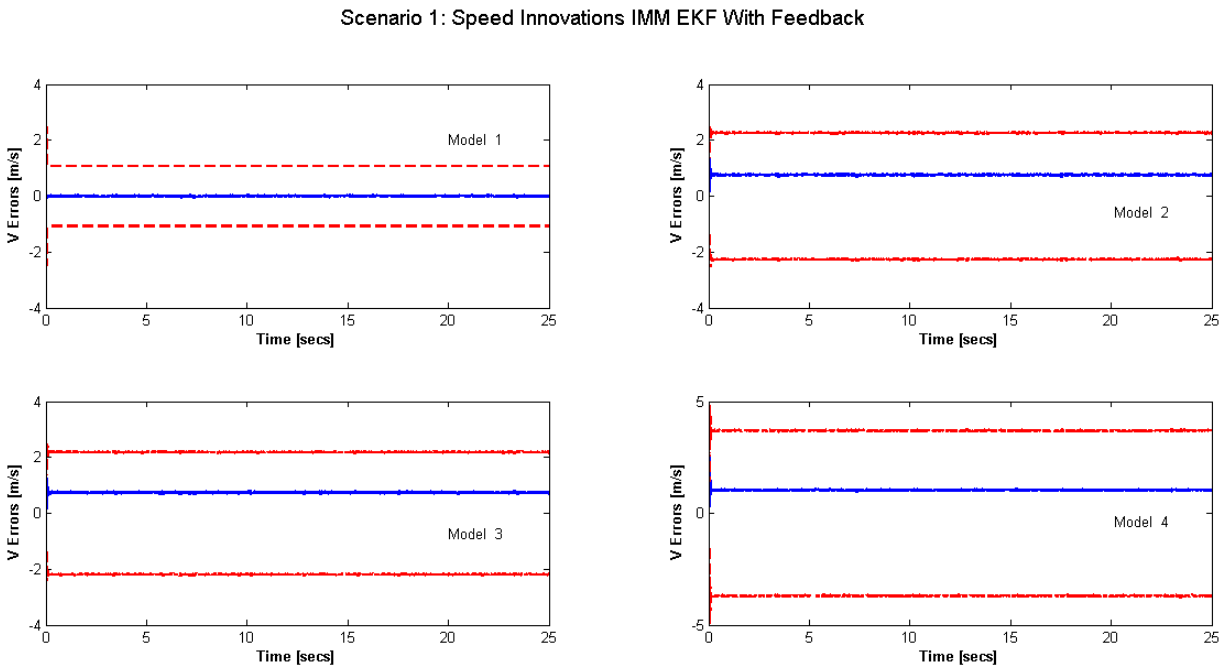

Figure 4.6: Scenario 1 - IMM EKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

Figure 4.7 and 4.8 are plots of the IMM UKF Speed errors with no feedback and with feedback respectively. The same behaviour is present as that seen in the IMM EKF case where mode 1 has the most confident estimates.

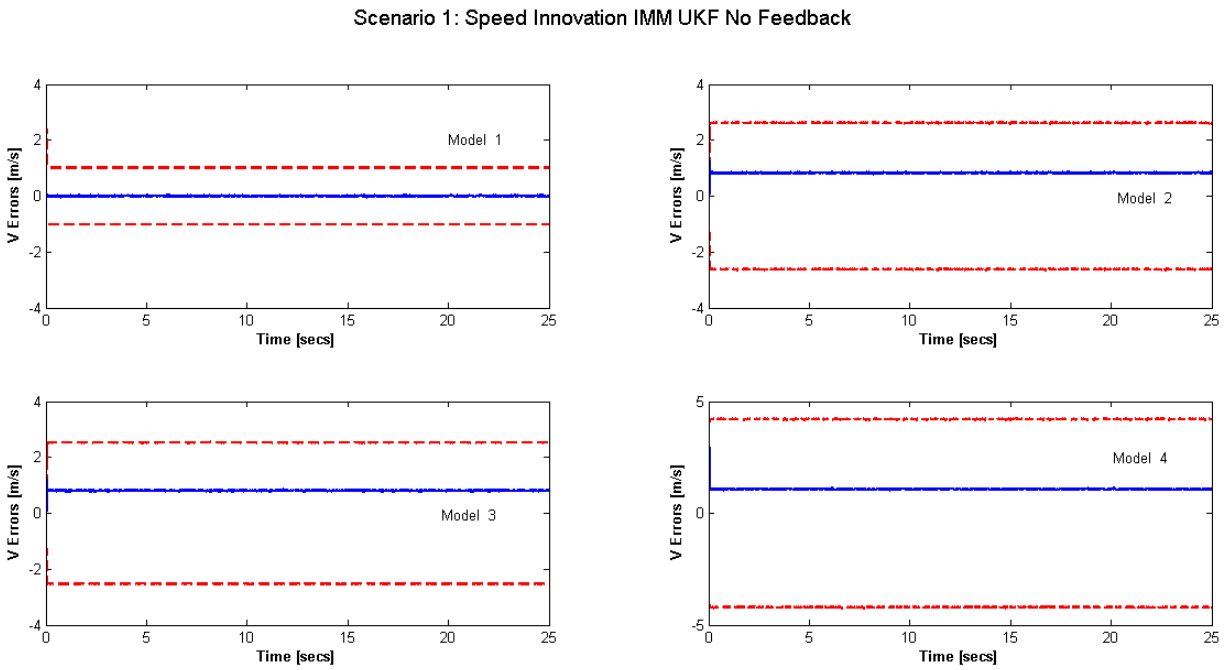

Figure 4.7: Scenario 1 - IMM UKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

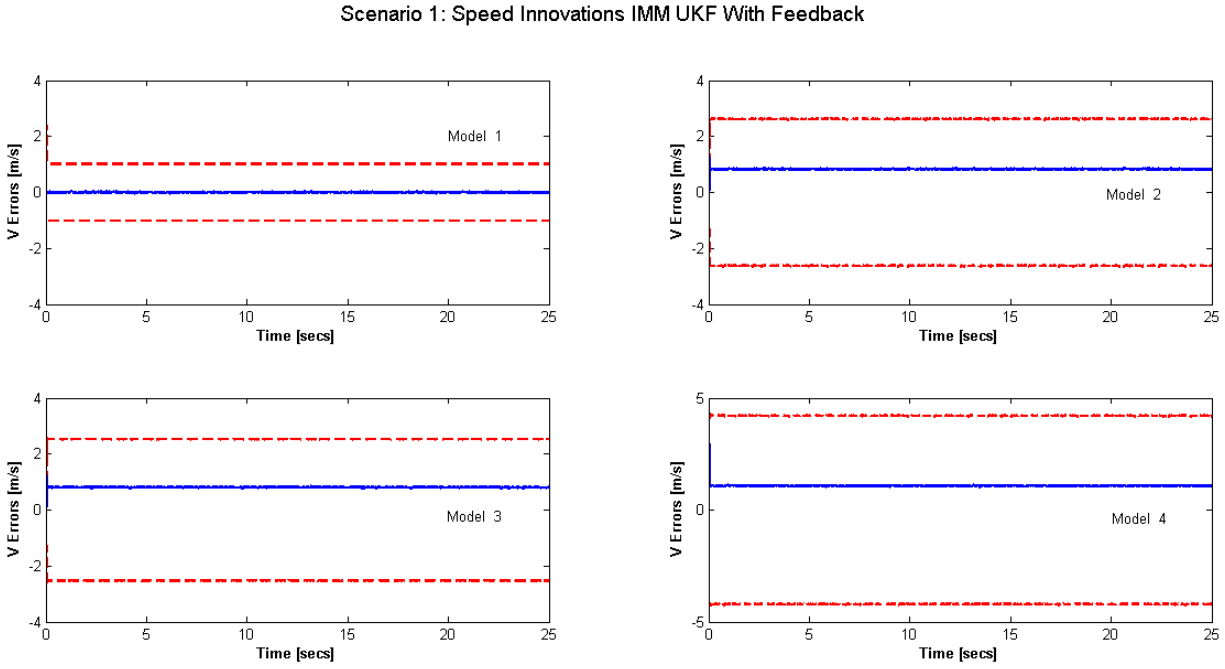

Figure 4.8: Scenario 1 - IMM UKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

4.5.1.2 Wheel Radius Estimates

Figures 4.9, 4.10, 4.11 and 4.12 are plots of the wheel radius estimates produced by each of the four filters respectively. The estimates are provided for both wheels with and without feedback to the controller. In both cases the filter does a very good job of estimating the radius of the tyres. The IMM filters initially have a higher error in the tyre estimate as the estimation is based on a mixture of all the models, however as can be seen in figure 4.11 and 4.12, it takes only one update for the IMM to reach the correct estimate.

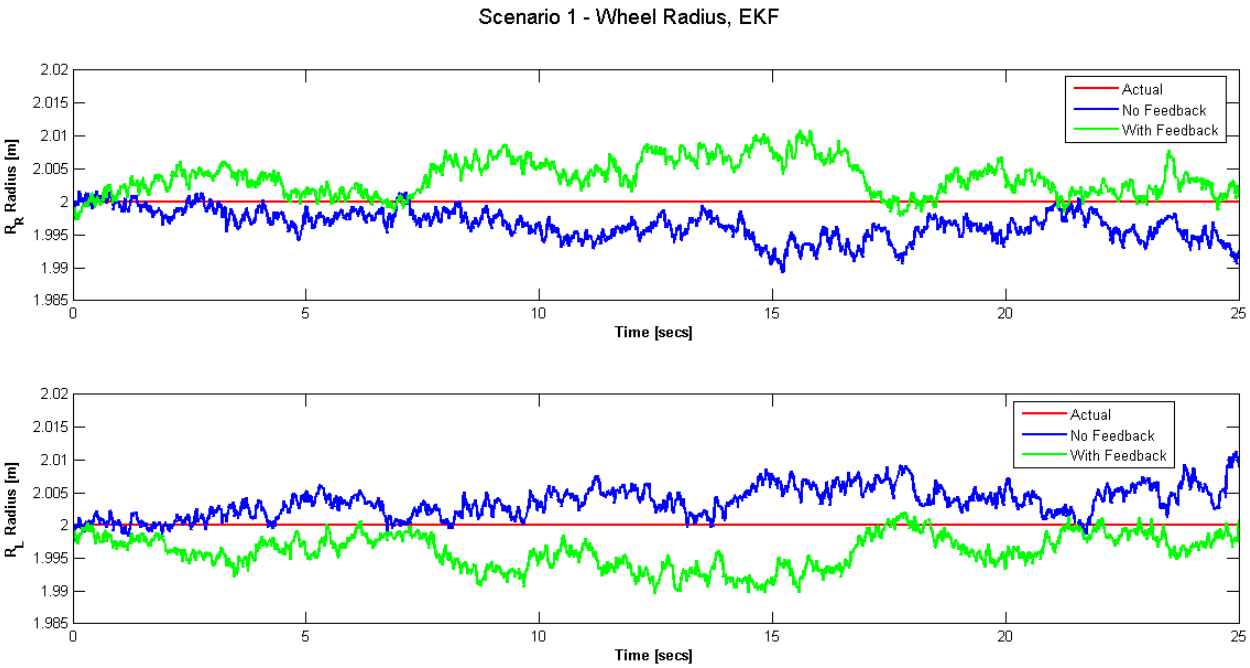

Figure 4.9: Scenario 1 - EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

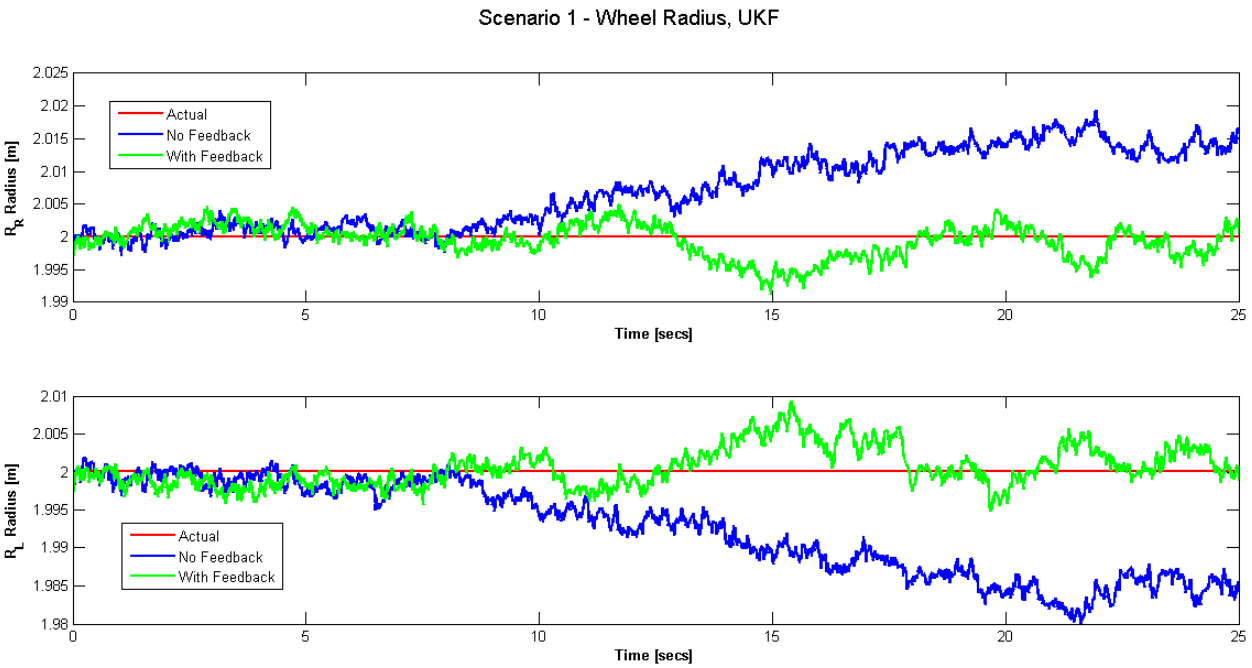

Figure 4.10: Scenario 1 - UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

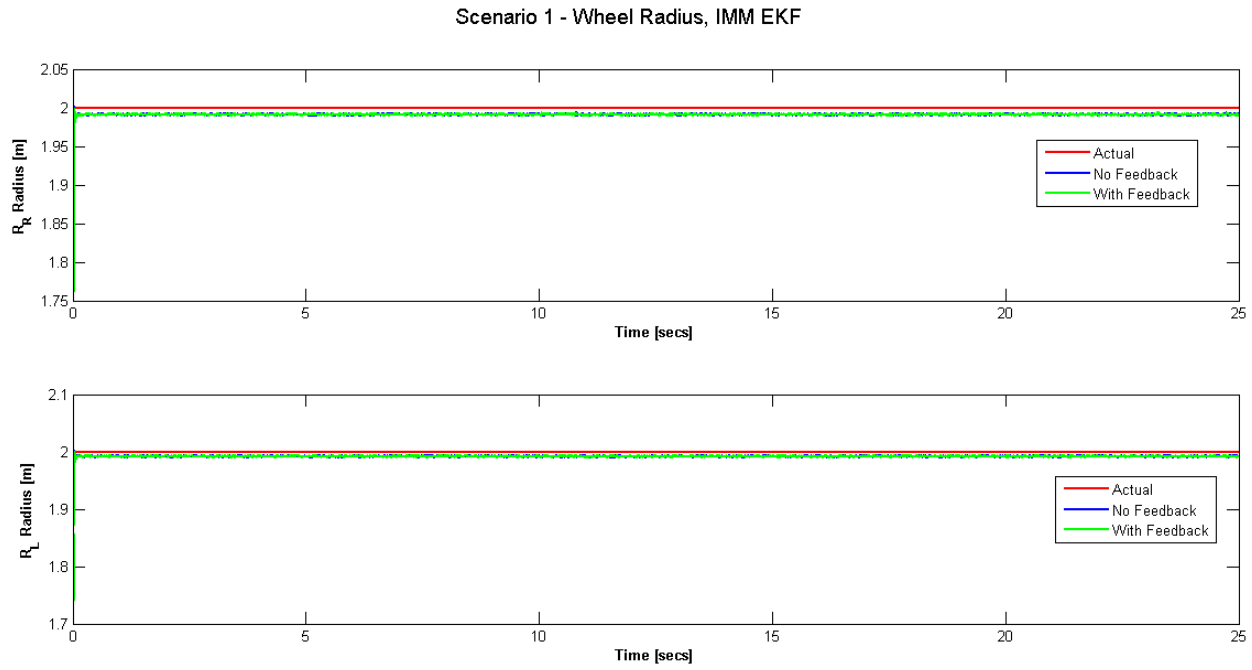

Figure 4.11: Scenario 1 - IMM EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

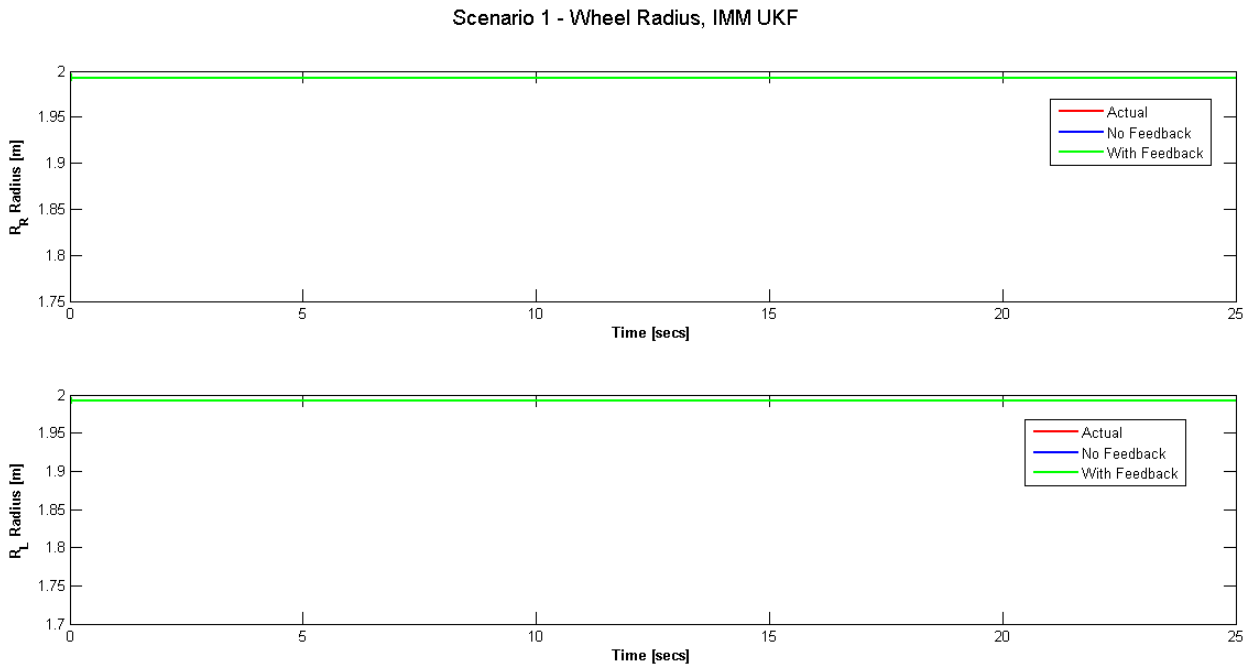

Figure 4.12: Scenario 1 - IMM UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

In figures 4.11 and 4.12 the estimates are very similar hence they are sitting one on top of the other making the red and blue lines invisible.

4.5.1.3 Wheel Speeds

The wheel speeds are given in figures 4.17, 4.18, 4.19 and 4.20 for each of the four filters respectively. For scenario 1 where there is no fault the wheel speeds for all four filters are very similar. In the case where feedback is provided the wheel speeds are quite noisy, a result of calculations based on noisy measurements which is a consequence of feedback.

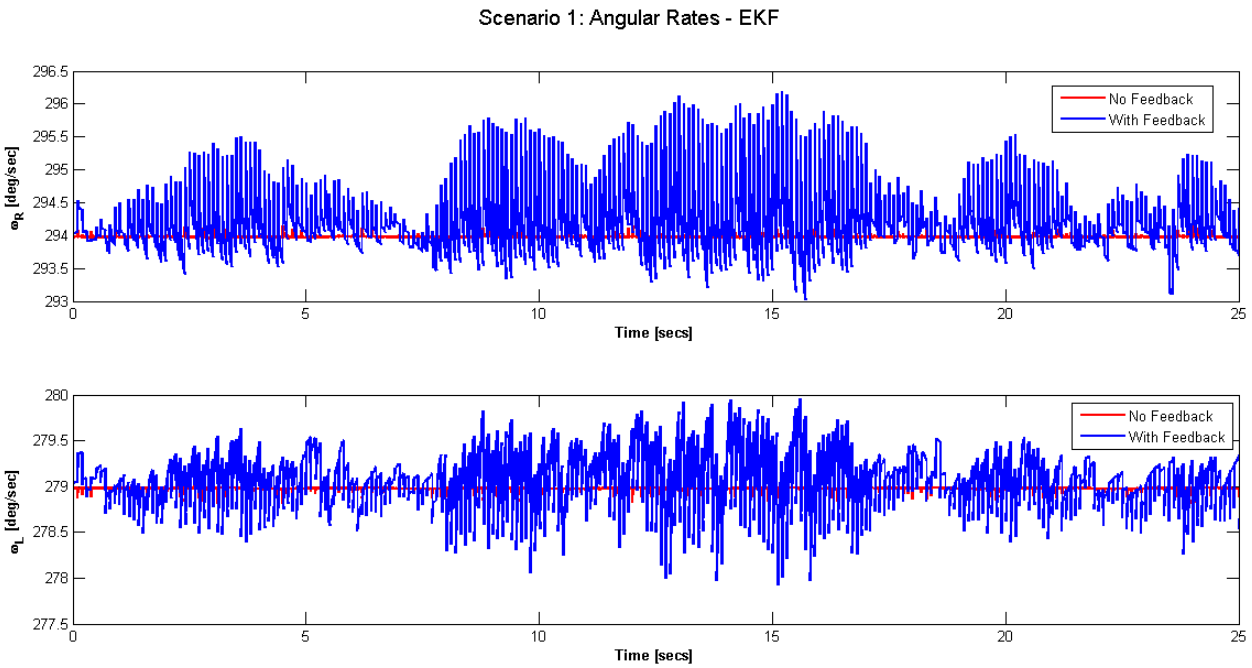

Figure 4.13: Scenario 1 - EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

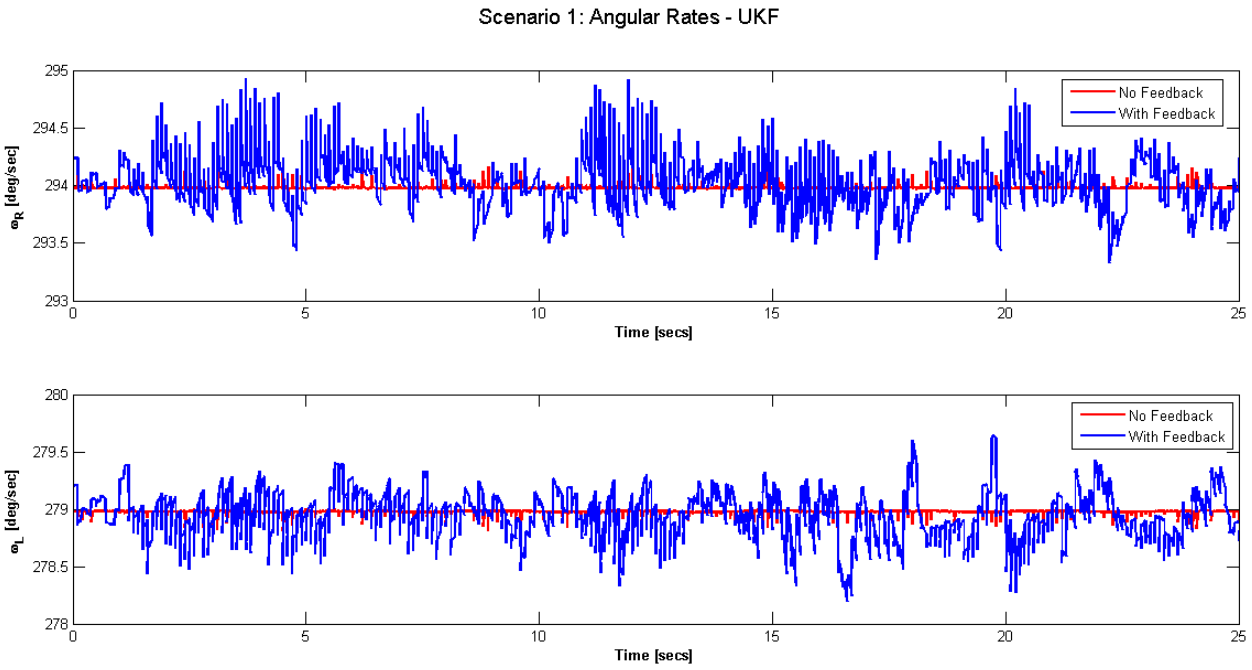

Figure 4.14: Scenario 1 - UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

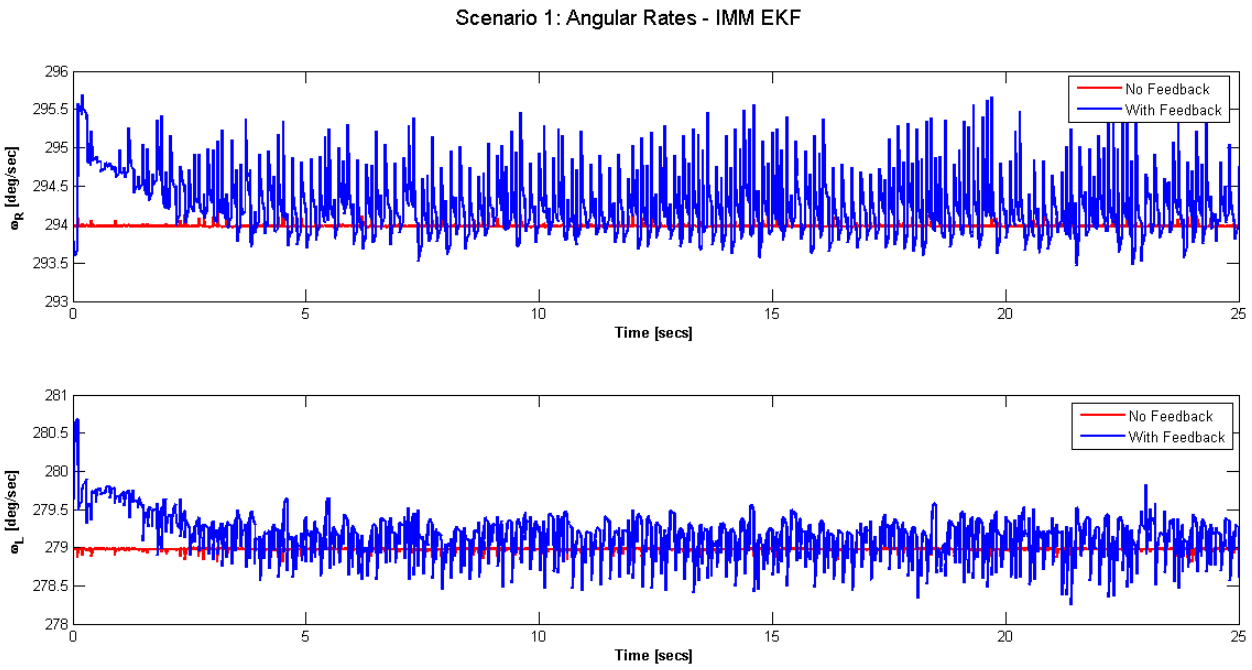

Figure 4.15: Scenario 1 - IMM EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

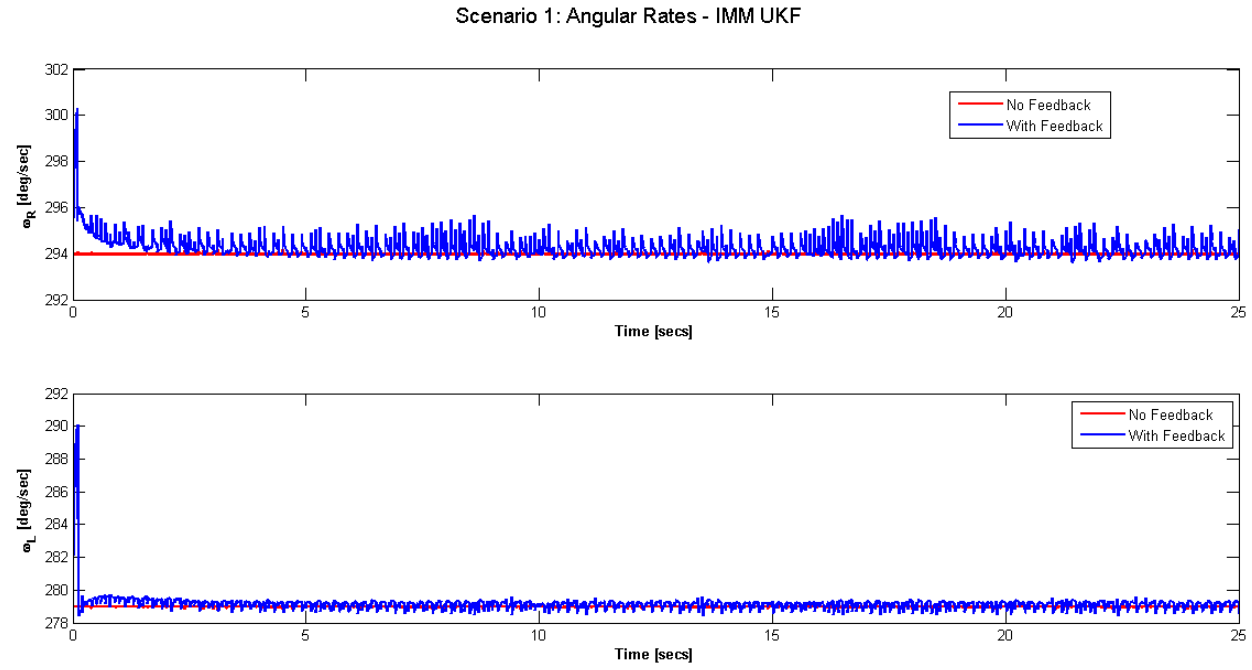

Figure 4.16: Scenario 1 - IMM UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

4.5.1.4 Trajectories

Plots of the robot trajectory are given in figures 4.17, 4.18, 4.19 and 4.20 for all four of the filters. The robot is seen to remain on the path with and without feedback which is to be expected in

the no fault case.

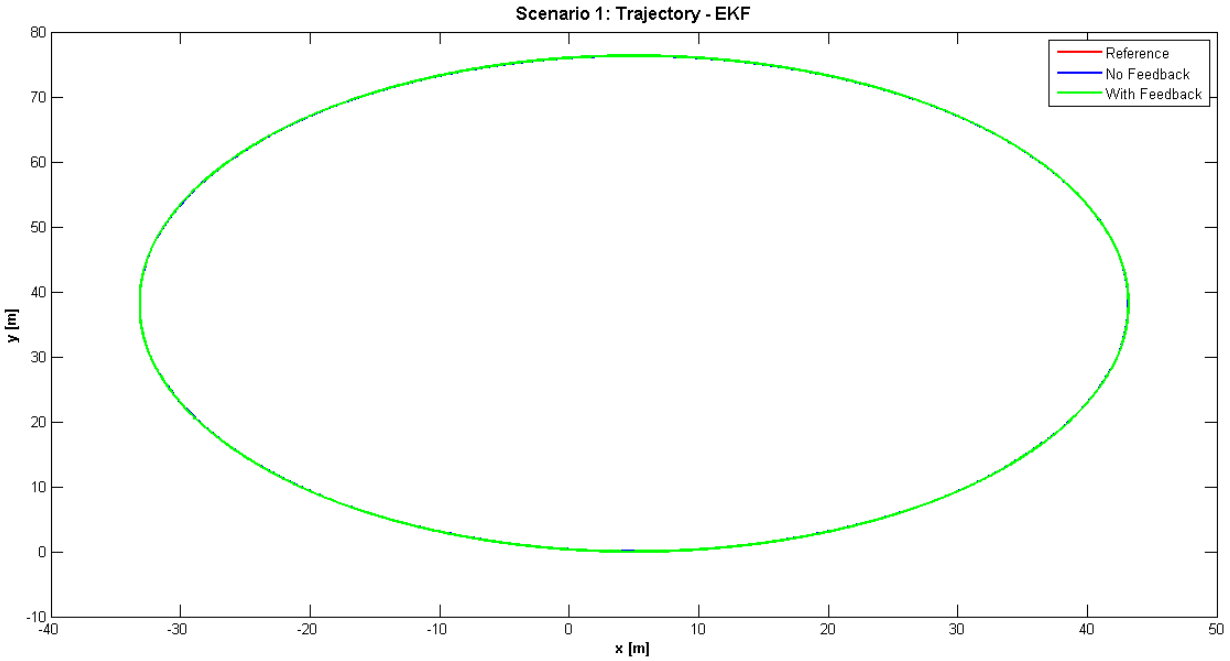

Figure 4.17: Scenario 1 - EKF Trajectory

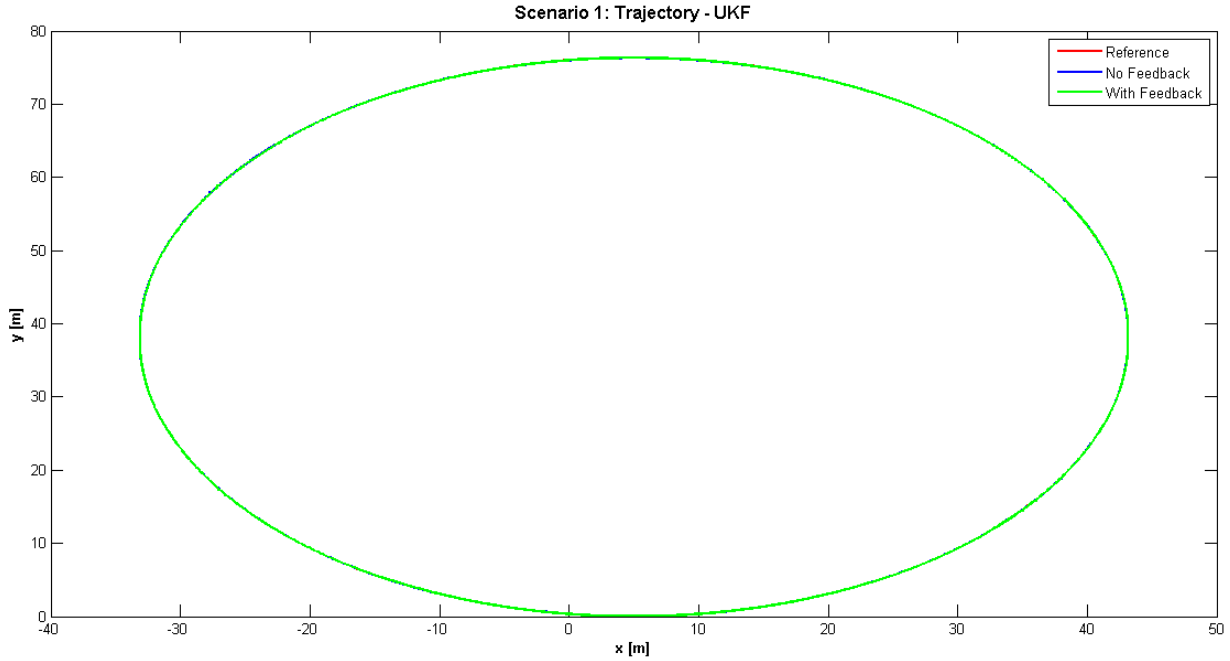

Figure 4.18: Scenario 1 - UKF Trajectory

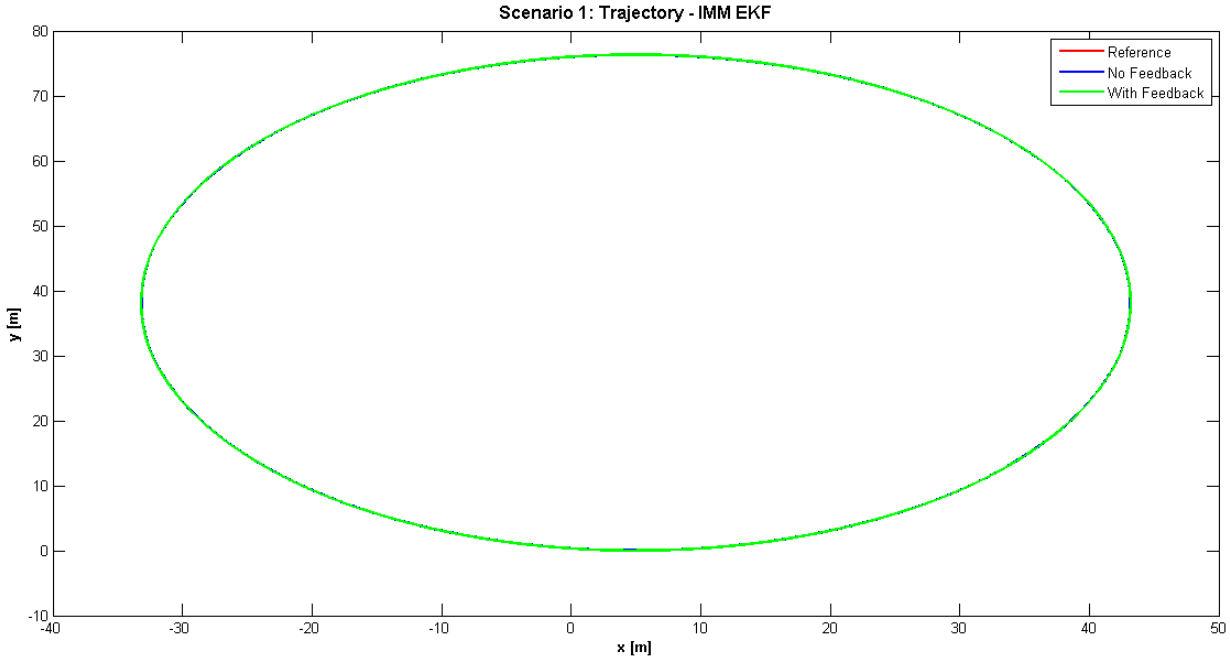

Figure 4.19: Scenario 1 - IMM EKF Trajectory

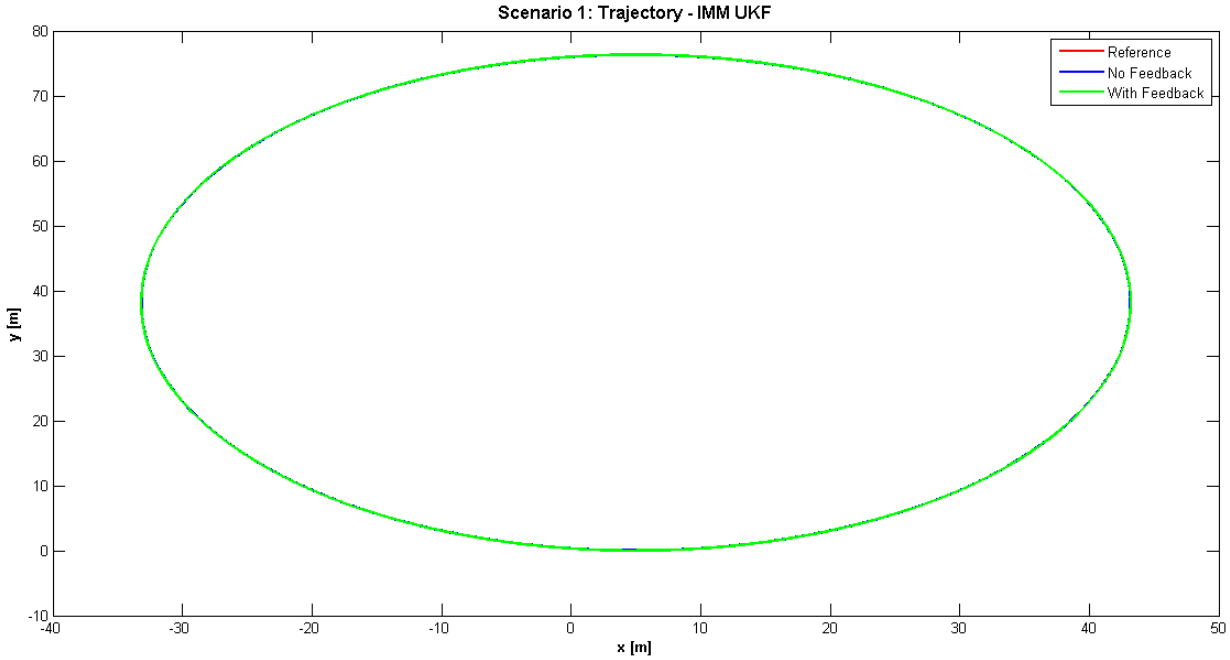

Figure 4.20: Scenario 1 - IMM UKF Trajectory

4.5.2 Scenario 2

4.5.2.1 Speed Errors (Innovations)

The innovations for the Speed are given in figures 4.21, 4.22, 4.23, 4.24, 4.25 and 4.26. The fault occurs at 10 secs into the run, and upon investigating the EKF and UKF innovation plots given in figures 4.21 and 4.22 respectively this can be seen from the peak change in the innovation curves. The corrections increase at the time the fault occurs and then settle again near zero once the correct estimate is reached. The IMM filters also show this peak change. Mode 3 is the correct match for scenario 2 and both of the IMM filters can be seen to find the correct mode immediately as mode 3 is the most confident in its estimate. Mode 3 and Mode 4 both hypothesise a failure in the left wheel of 50% which is why after the occurrence of the fault the uncertainties do not increase. However because mode 1 and mode 2 do not hypothesise a fault in the left wheel the uncertainties can be seen to increase once the fault has occurred. Mode 3 is seen to have more of a decrease in uncertainty than mode 4 after the fault occurrence because mode 4 hypothesises that both wheels are punctured whereas mode 3 predicts the puncture of only the left wheel. In both cases the EKF and UKF exhibited similar performance. Another point to note is that in the single filter cases feedback did not have much of an effect on the innovations. However, in the case of the IMM filters the results show that with feedback the filter errors do not grow as rapidly between updates. The errors are seen to grow very quickly when no feedback is present.

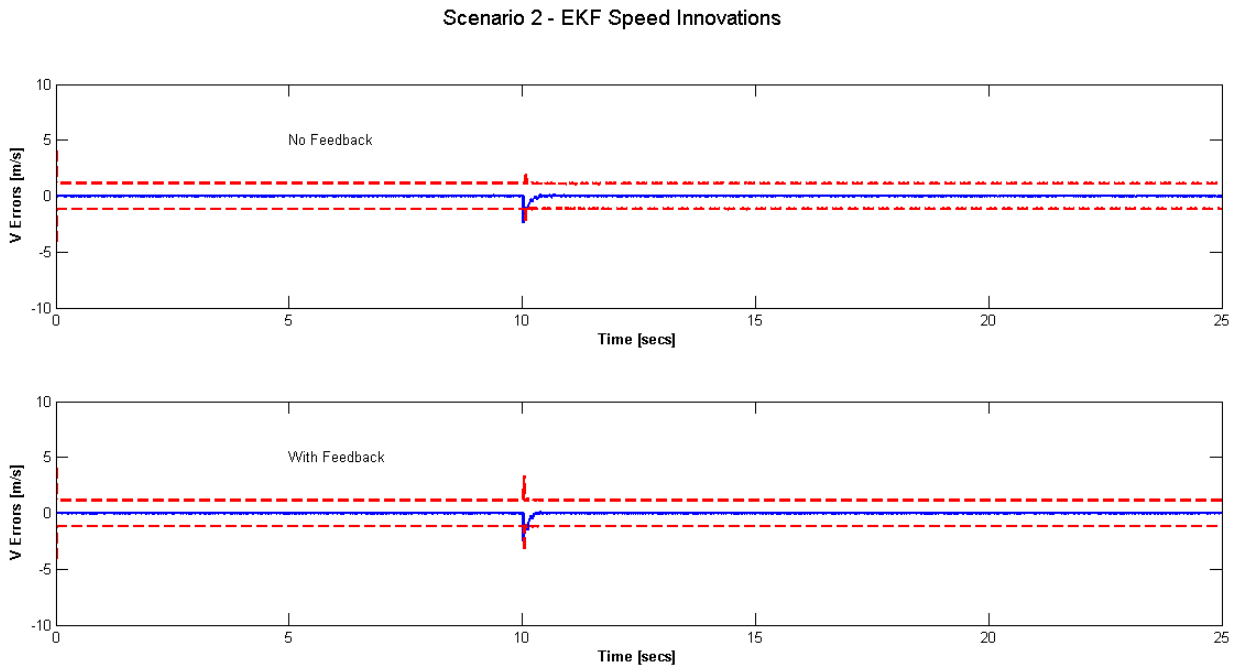

Figure 4.21: Scenario 2 - EKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

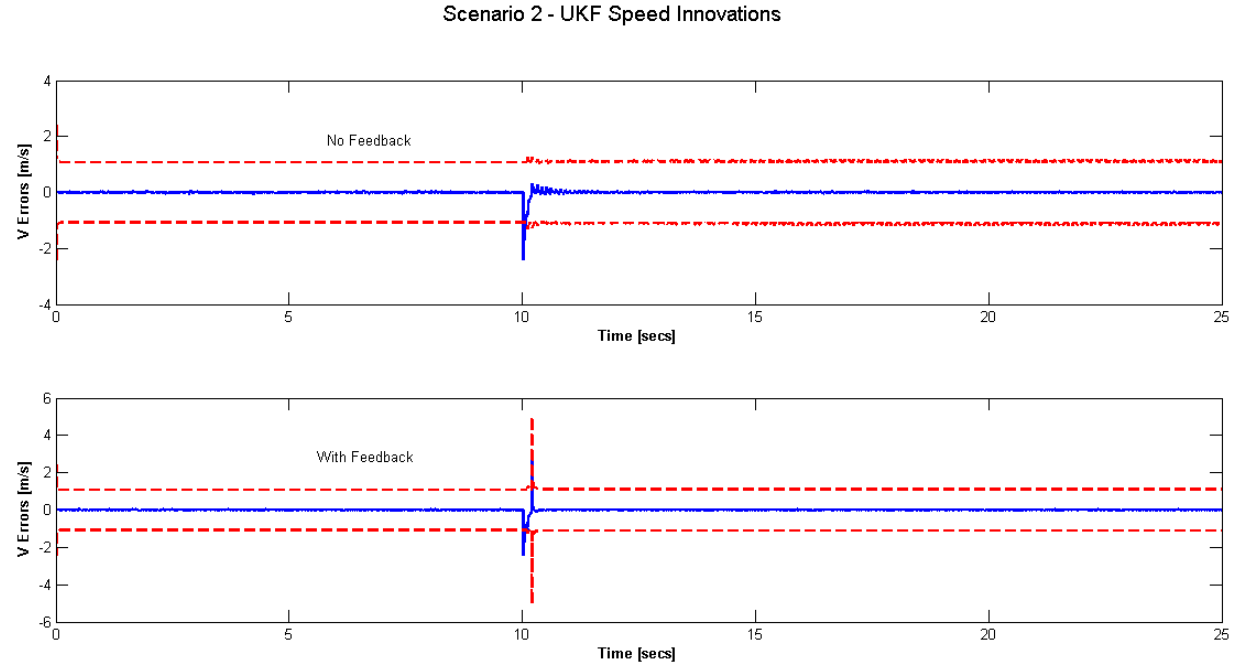

Figure 4.22: Scenario 2 - UKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

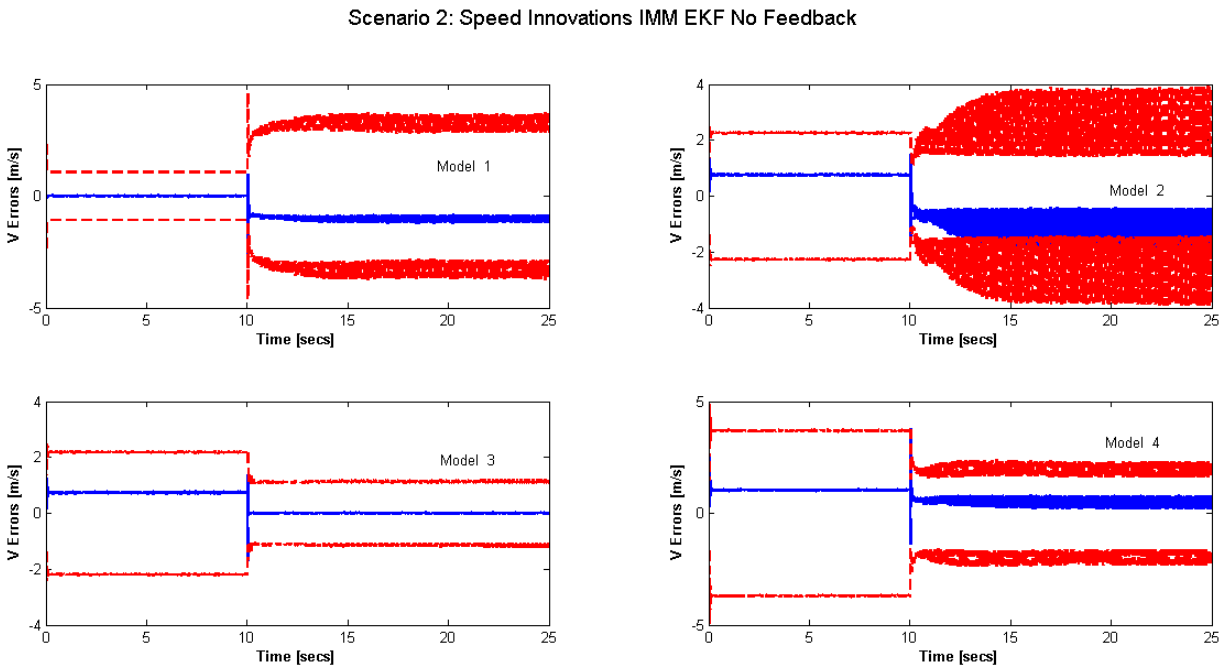

Figure 4.23: Scenario 2 - IMM EKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

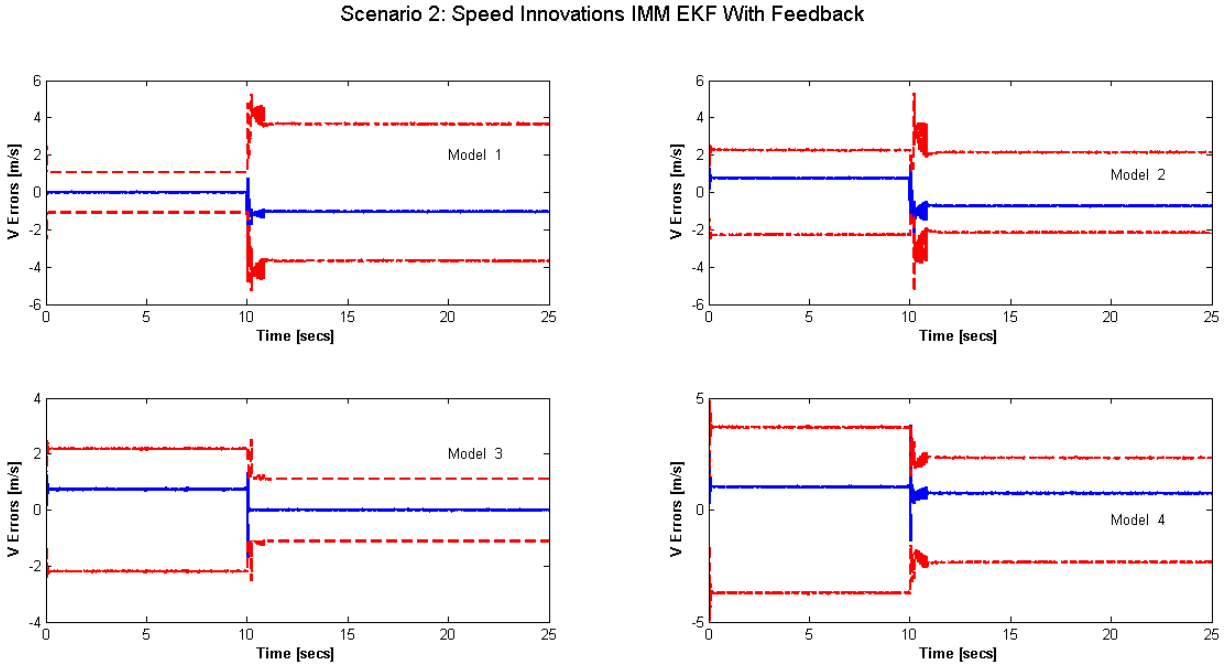

Figure 4.24: Scenario 2 - IMM EKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

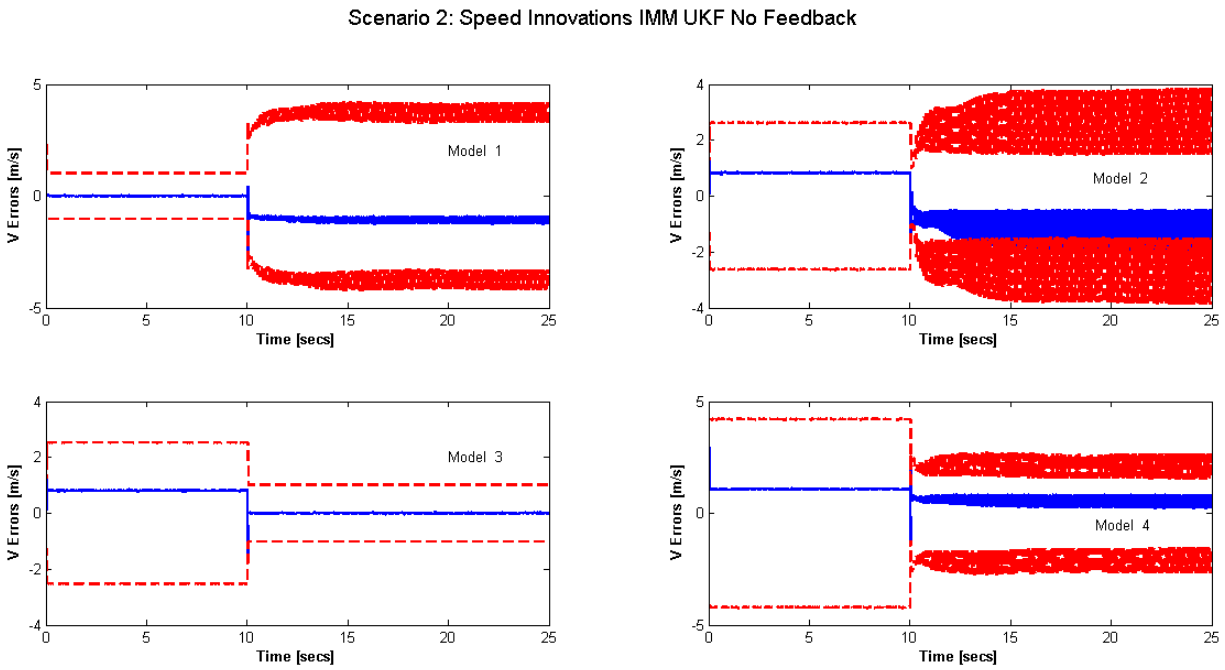

Figure 4.25: Scenario 2 - IMM UKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

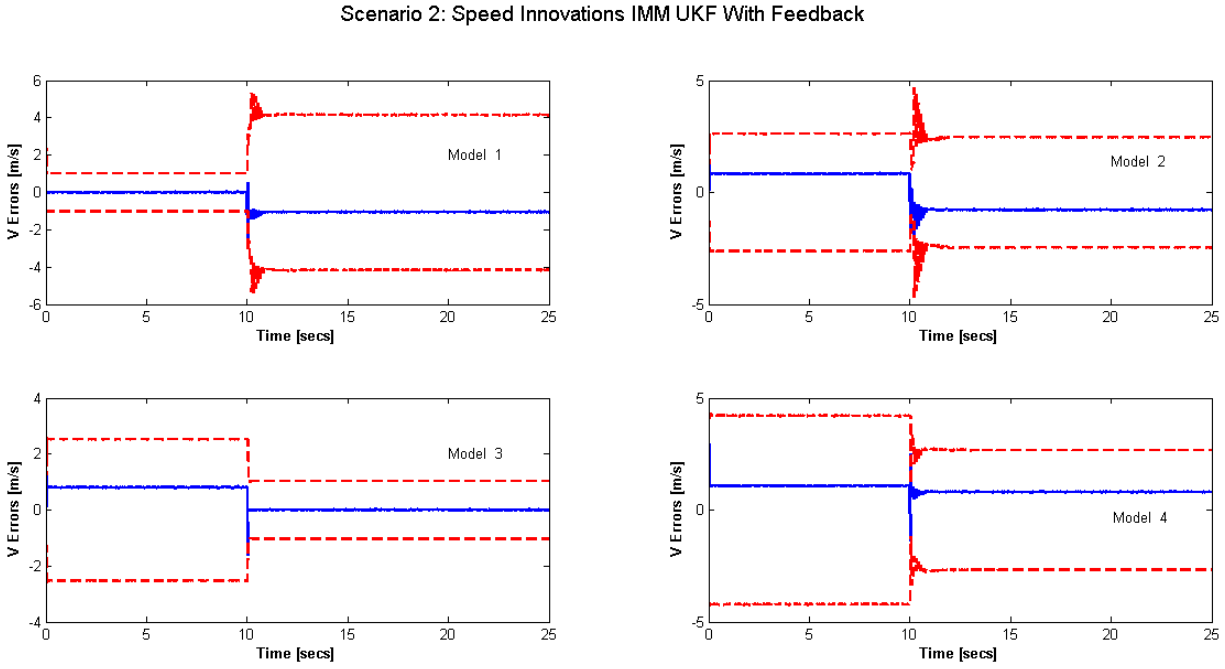

Figure 4.26: Scenario 2 - IMM UKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

4.5.2.2 Wheel Radius Estimates

The wheel radius estimates for EKF are given in figure 4.27, and for UKF in figure 4.28. The results show that the UKF estimates are closer to the actual wheel radius compared to those produced by the EKF. Turning on feedback results in the filters reaching a steady estimate faster when compared to the no feedback case. The IMM filters produce better estimates than the single filters as can be seen from figures 4.29 and 4.30 with feedback providing very little improvement on the estimate.

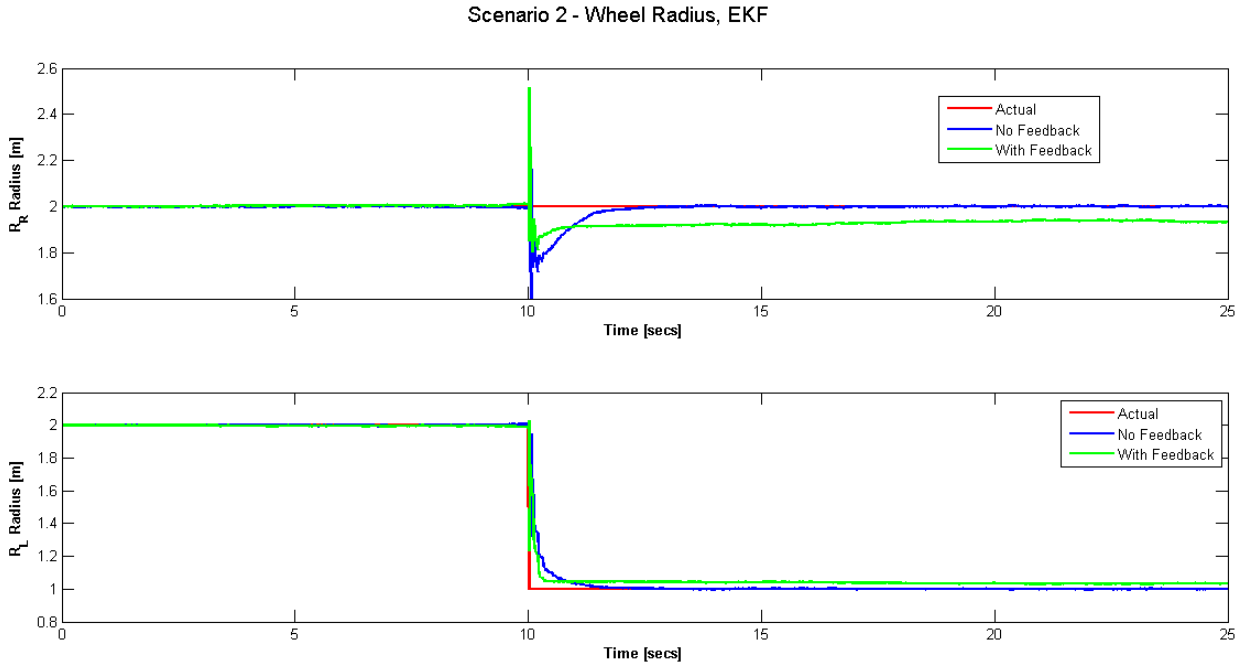

Figure 4.27: Scenario 2 - EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

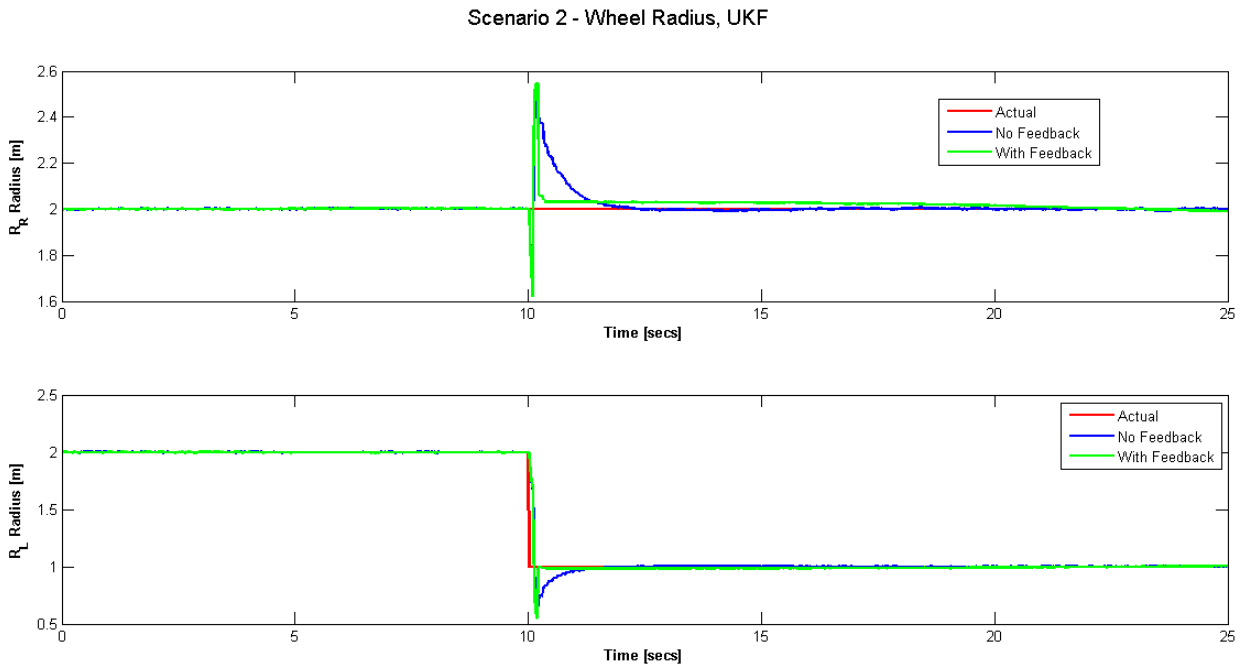

Figure 4.28: Scenario 2 - UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

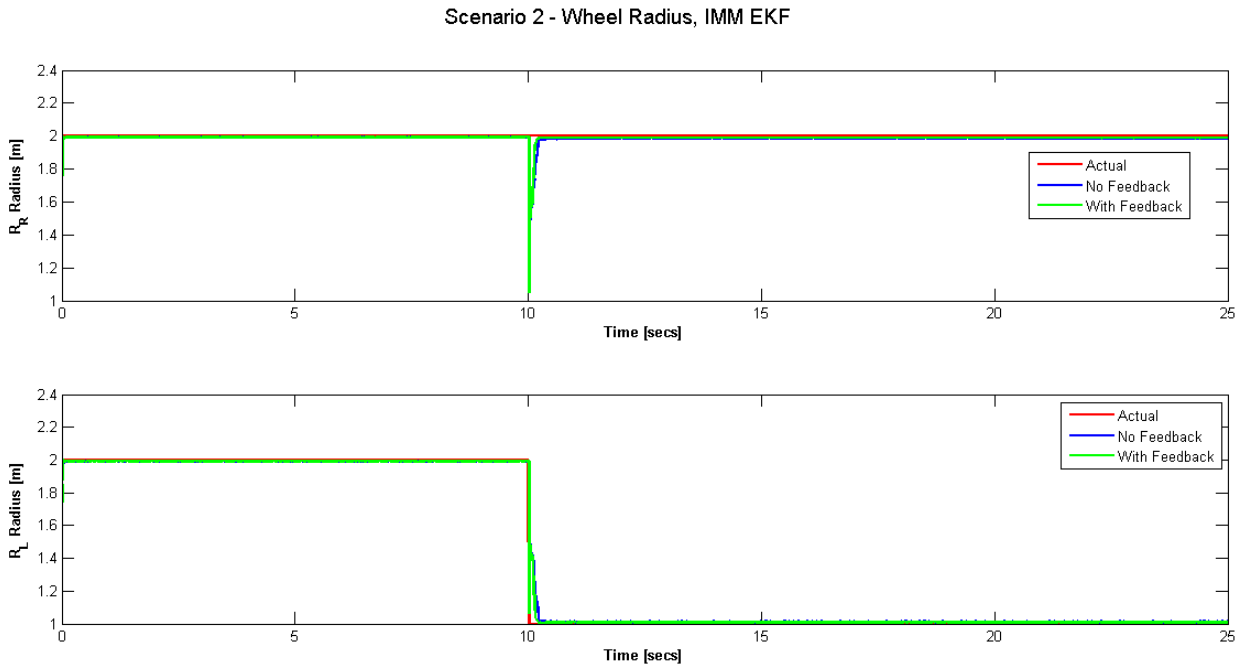

Figure 4.29: Scenario 2 - IMM EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

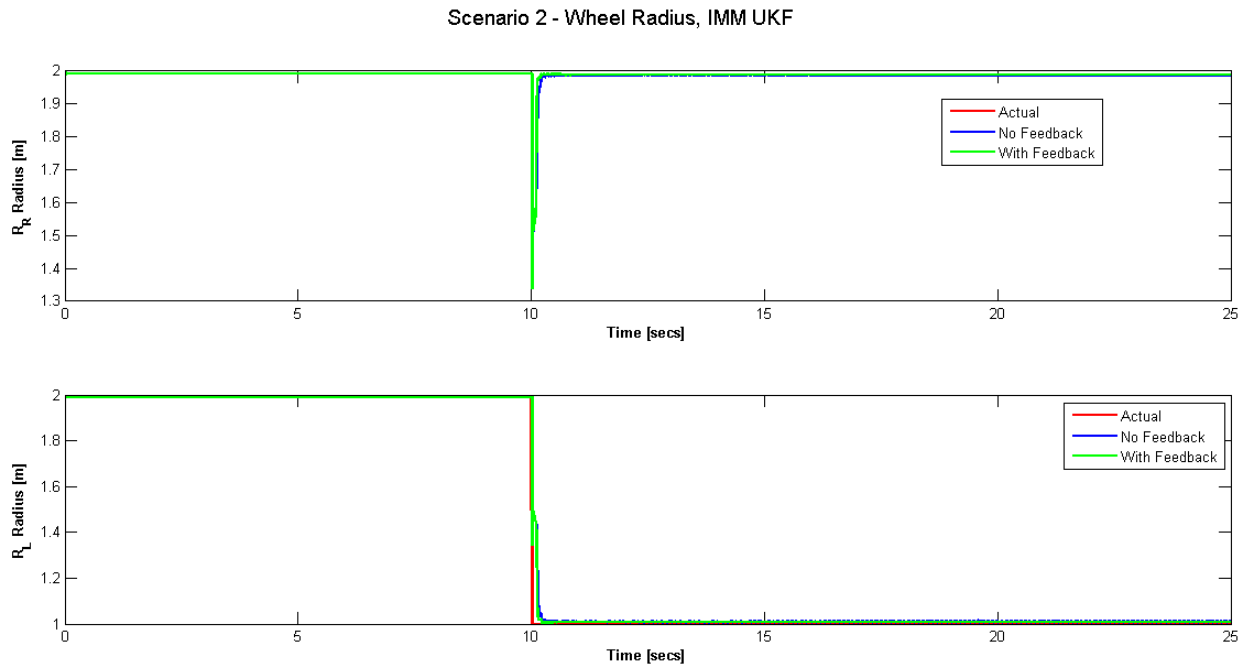

Figure 4.30: Scenario 2 - IMM UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

4.5.2.3 Wheel Speeds

The wheel speeds are given in figures 4.31, 4.32, 4.33 and 4.34. The results for all four filters present the same trends. Without feedback there is much more activity present compared to turning on the feedback. Once the fault occurs the robot yaws to the side with the punctured wheel and demands a faster speed to compensate for the loss in radius.

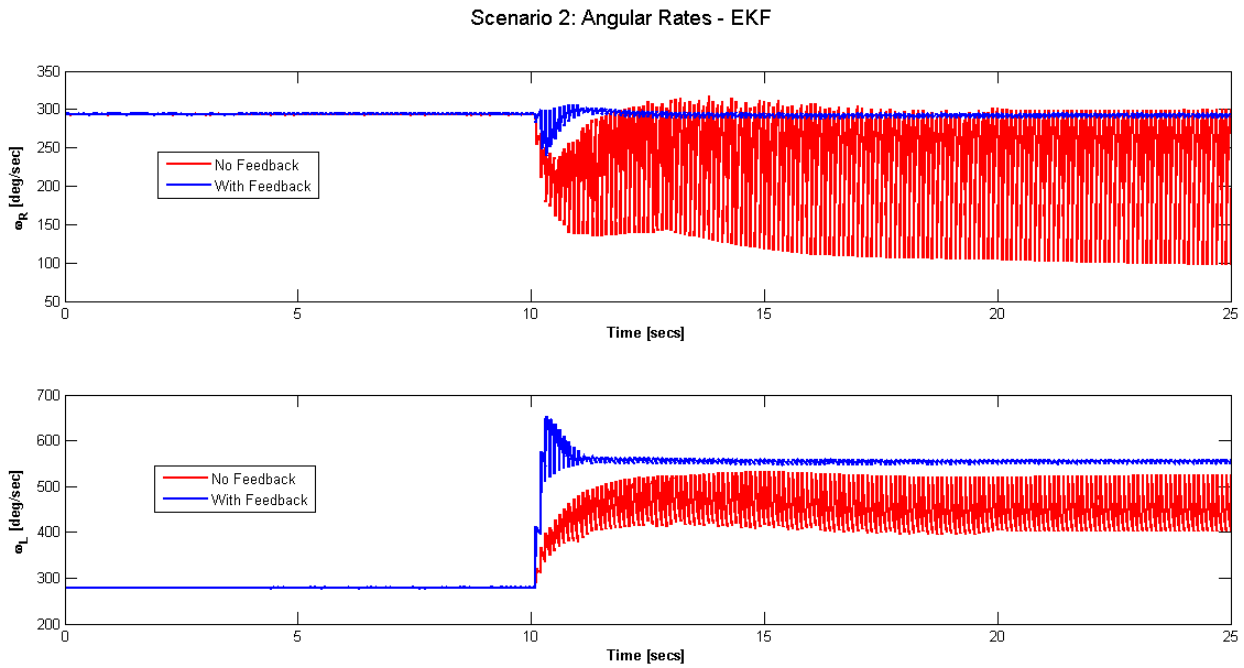

Figure 4.31: Scenario 2 - EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

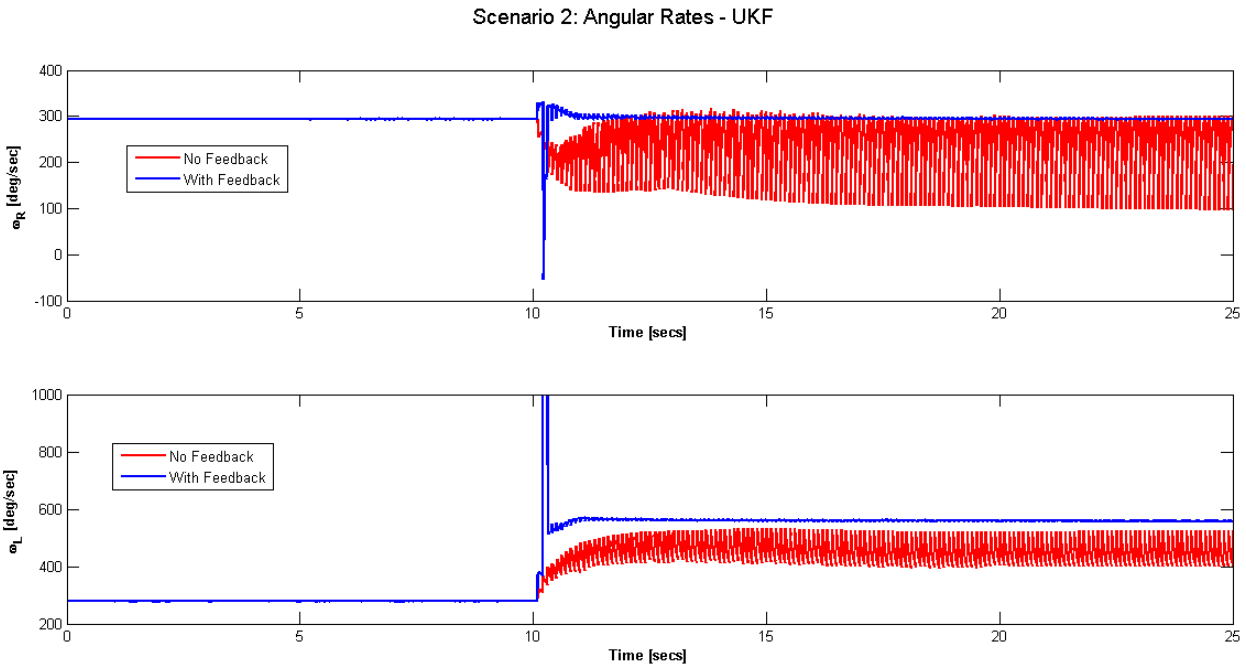

Figure 4.32: Scenario 2 - UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

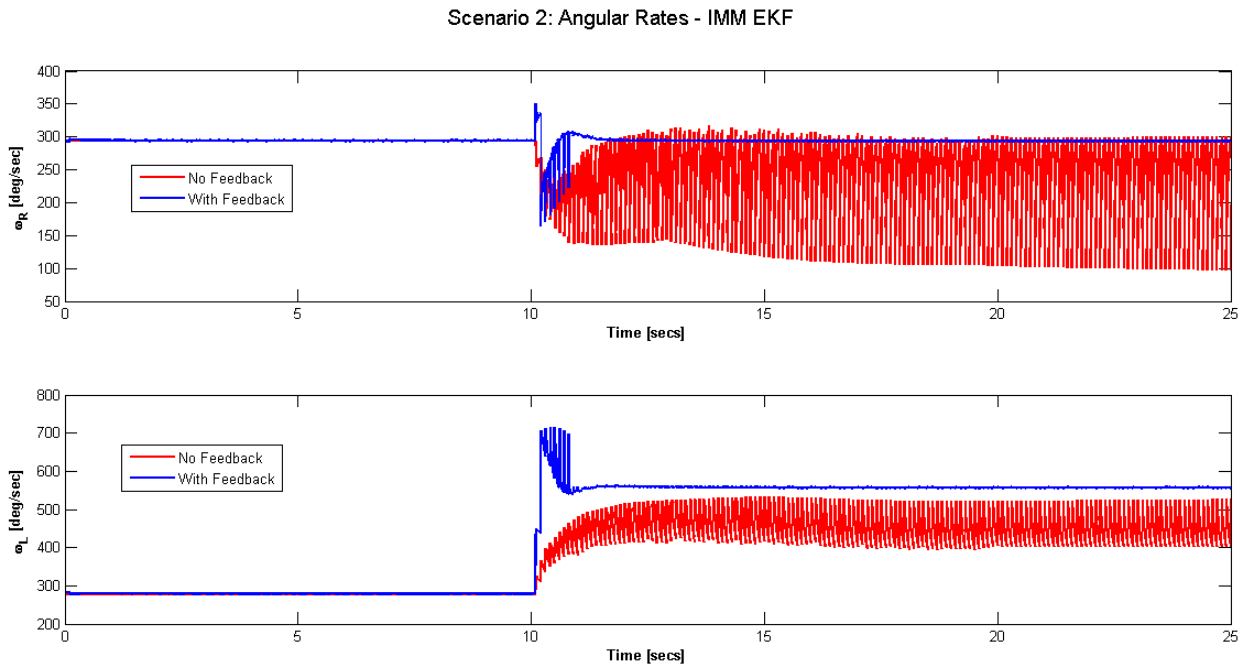

Figure 4.33: Scenario 2 - IMM EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

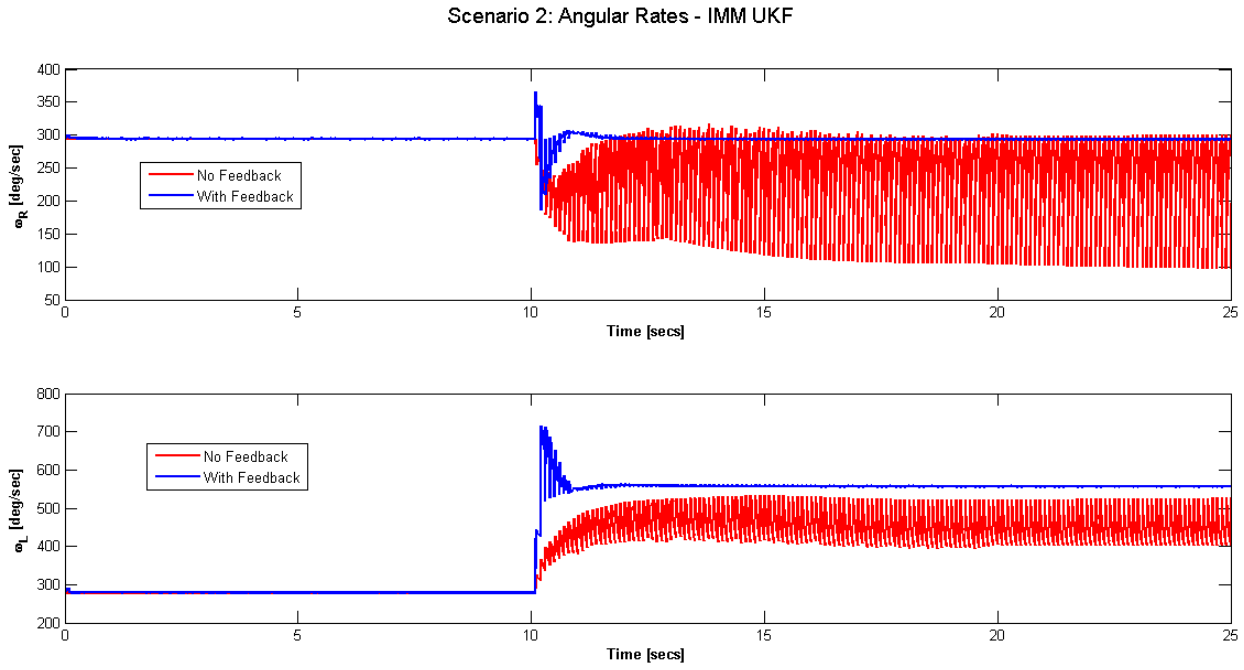

Figure 4.34: Scenario 2 - IMM UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

4.5.2.4 Trajectories

The trajectory for the filters given in figures 4.35, 4.36, 4.37 and 4.38 all show that the robot was only able to remain on the path if feedback from the filter was provided to reconfigure the con-

troller. Although the filters' estimates without feedback were excellent, without reconfiguration of the controller the robot could not be made to follow the desired path.

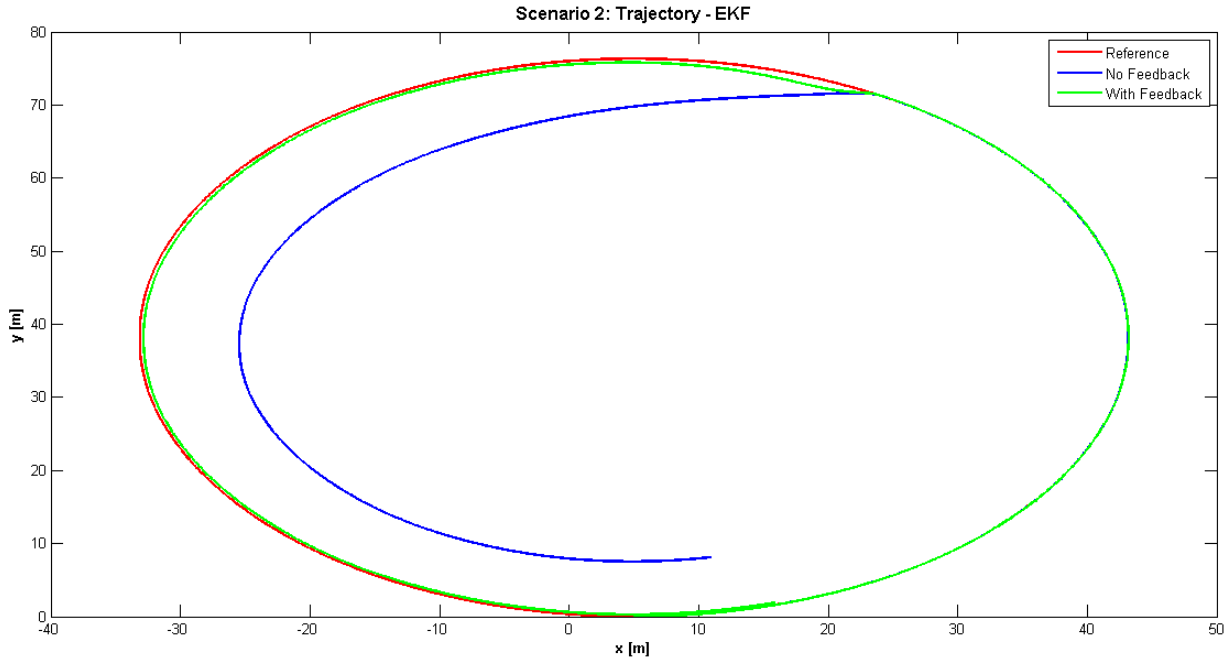

Figure 4.35: Scenario 2 - EKF Trajectory

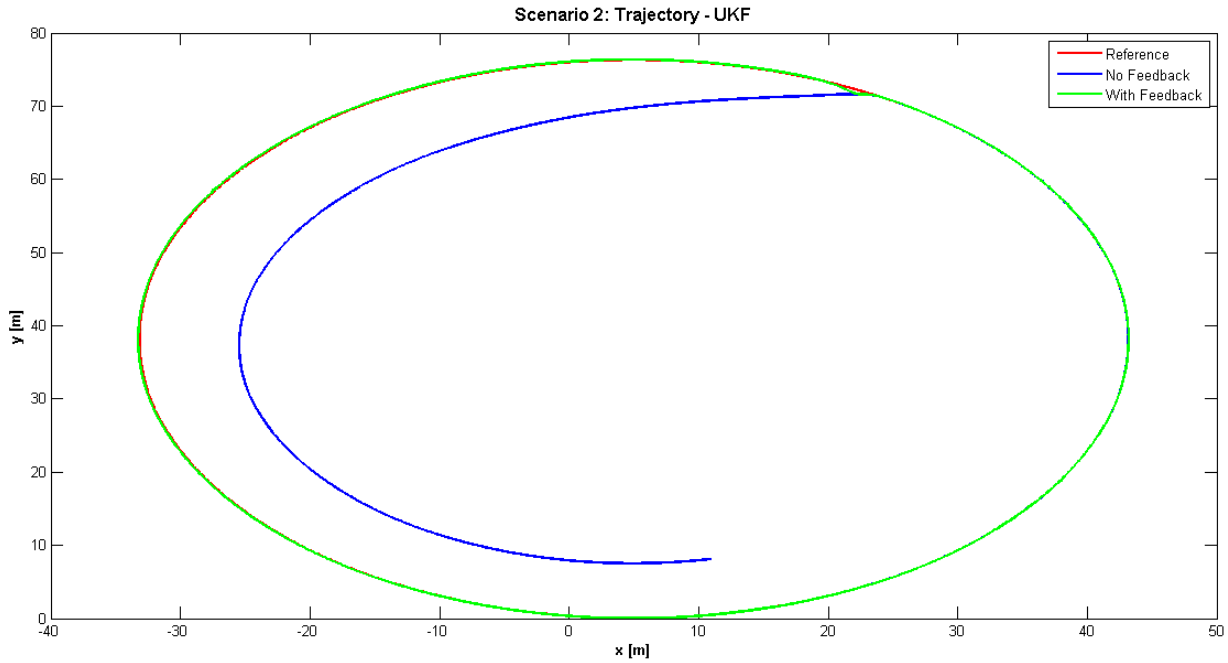

Figure 4.36: Scenario 2 - UKF Trajectory

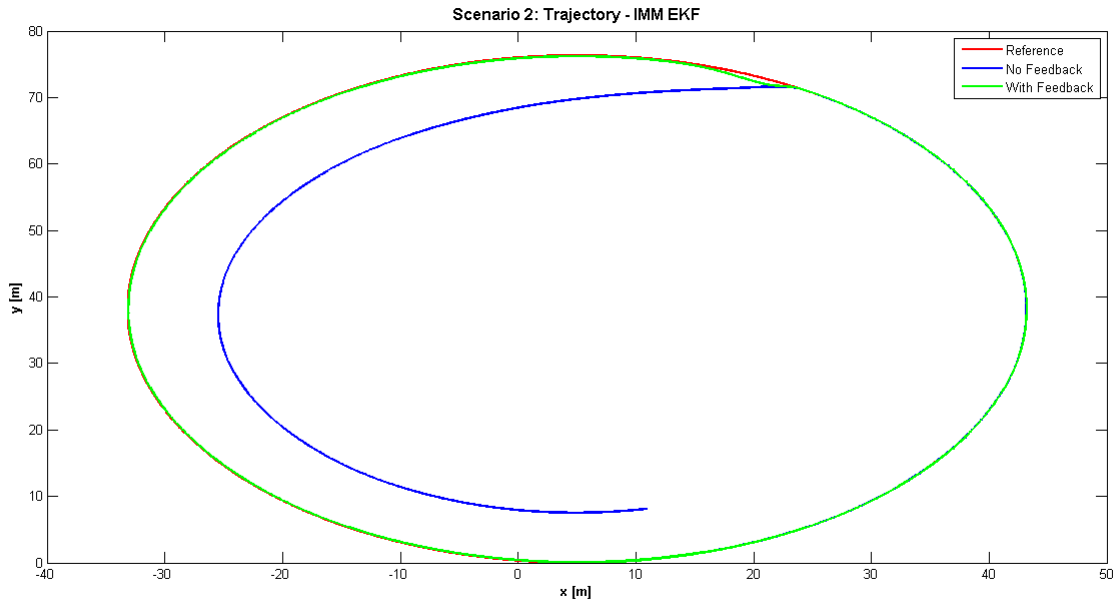

Figure 4.37: Scenario 2 - IMM EKF Trajectory

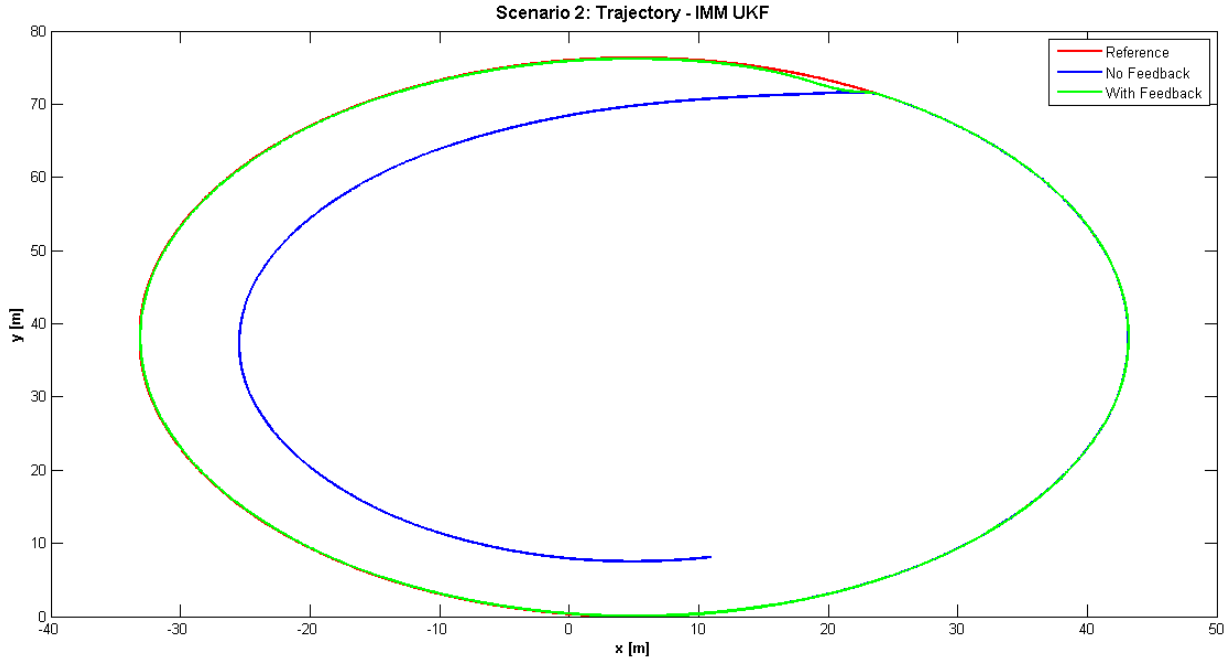

Figure 4.38: Scenario 2 - IMM UKF Trajectory

4.5.3 Scenario 3

4.5.3.1 Speed Errors (Innovations)

Figures 4.39, 4.40, 4.41, 4.42, 4.43 and 4.44 show plots of the speed innovations for scenario 3 for all four filters with and without feedback. All plots clearly indicate, from the sudden changes

in innovations, the detection of both faults, left wheel at 5 secs and right wheel at 10 secs. The EKF innovations were found to be consistent, however the innovations produced by the UKF show that, with feedback, the innovation uncertainty begins to grow rapidly between updates whereas without feedback the uncertainty remains constant. The IMM filters show that after 5 secs model 3 is the best model. However once the second fault occurs the filters do an excellent job of recognising that mode 4 is the correct match and uncertainties in mode 4 are seen to decrease. Again with no feedback the uncertainties on the IMM filters grow rapidly between updates and many more corrections are required.

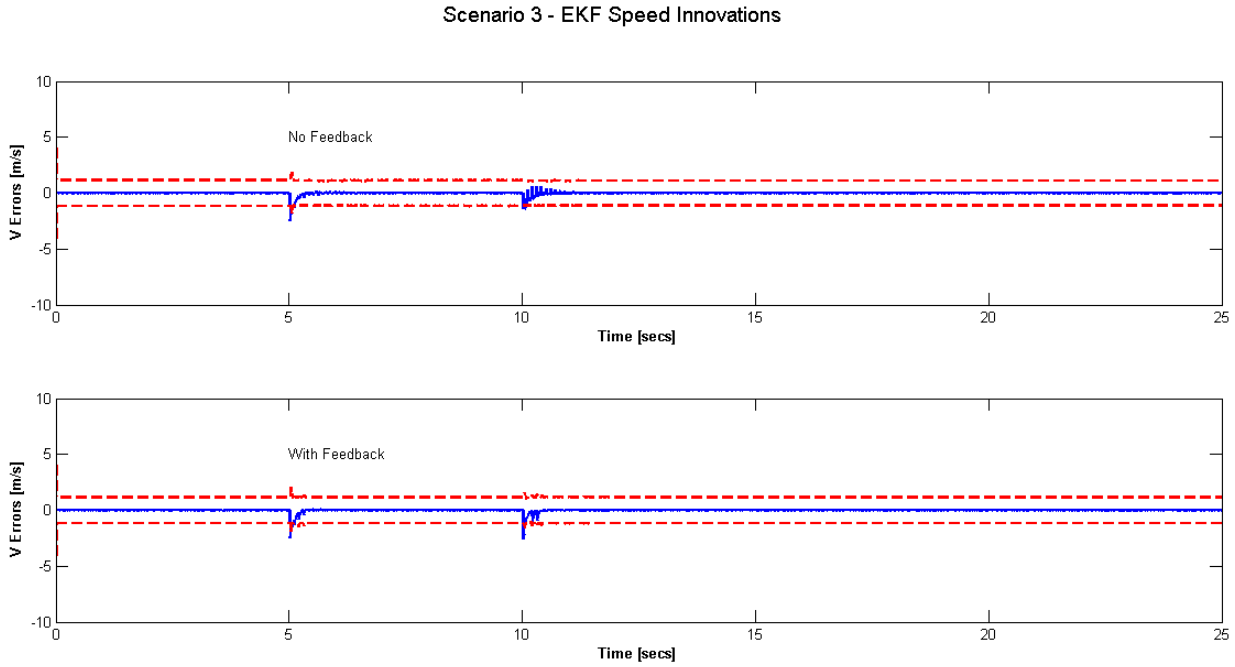

Figure 4.39: Scenario 3 - EKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

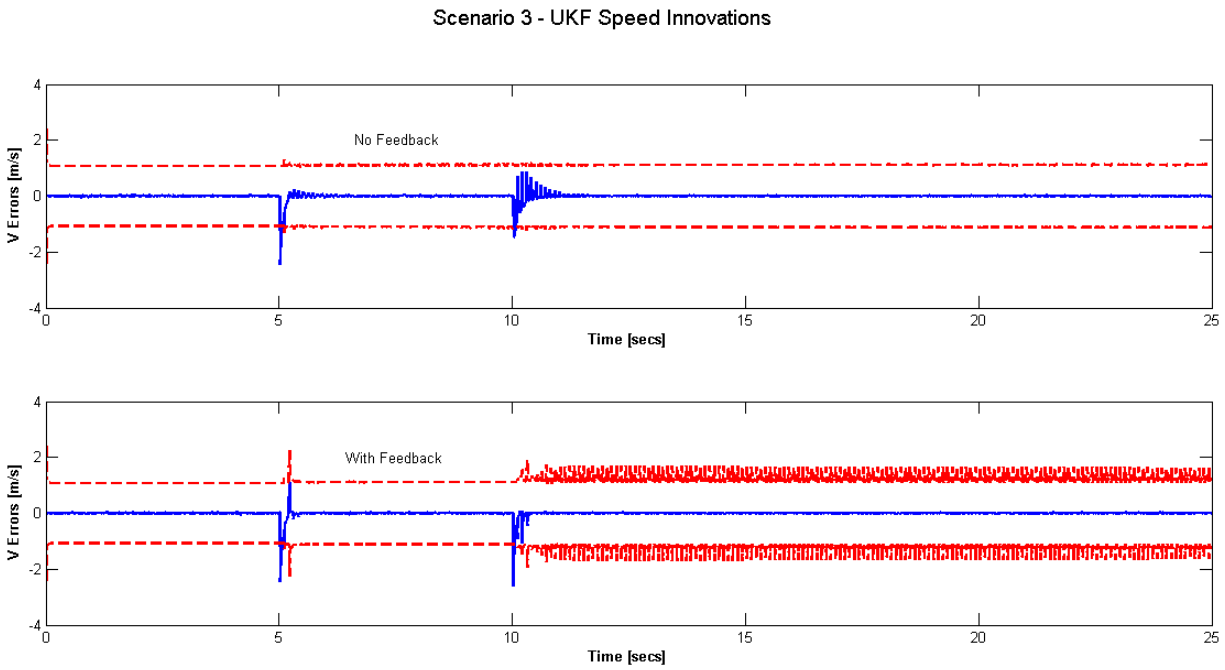

Figure 4.40: Scenario 3 - UKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

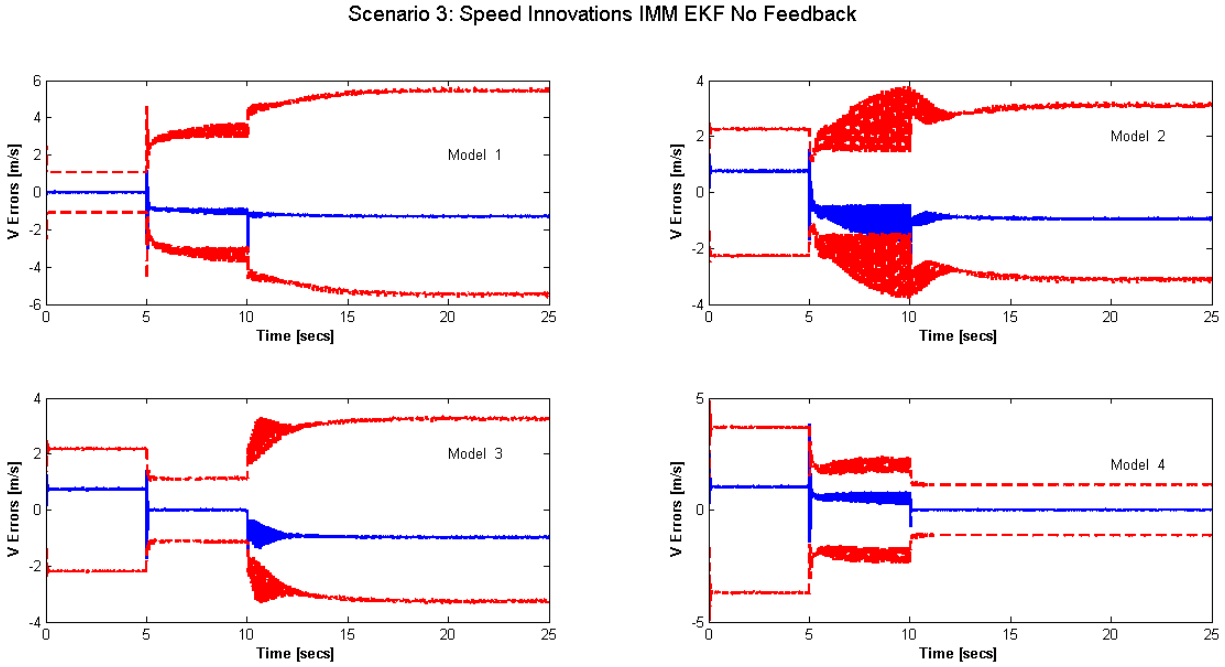

Figure 4.41: Scenario 3 - IMM EKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

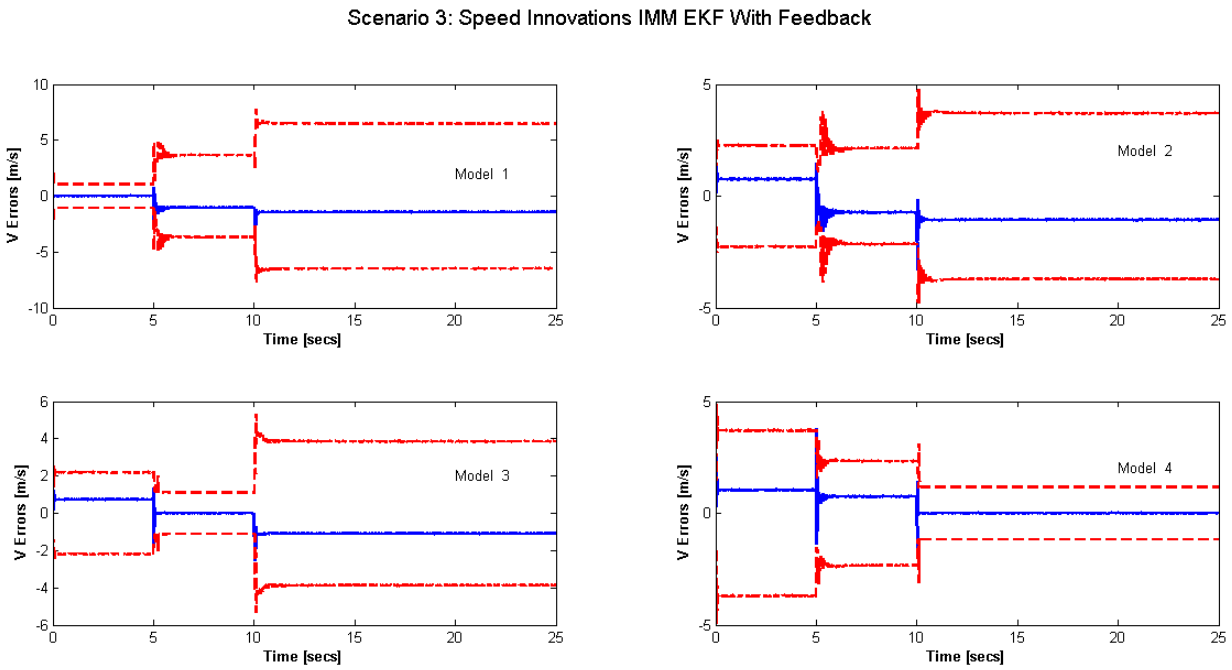

Figure 4.42: Scenario 3 - IMM EKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

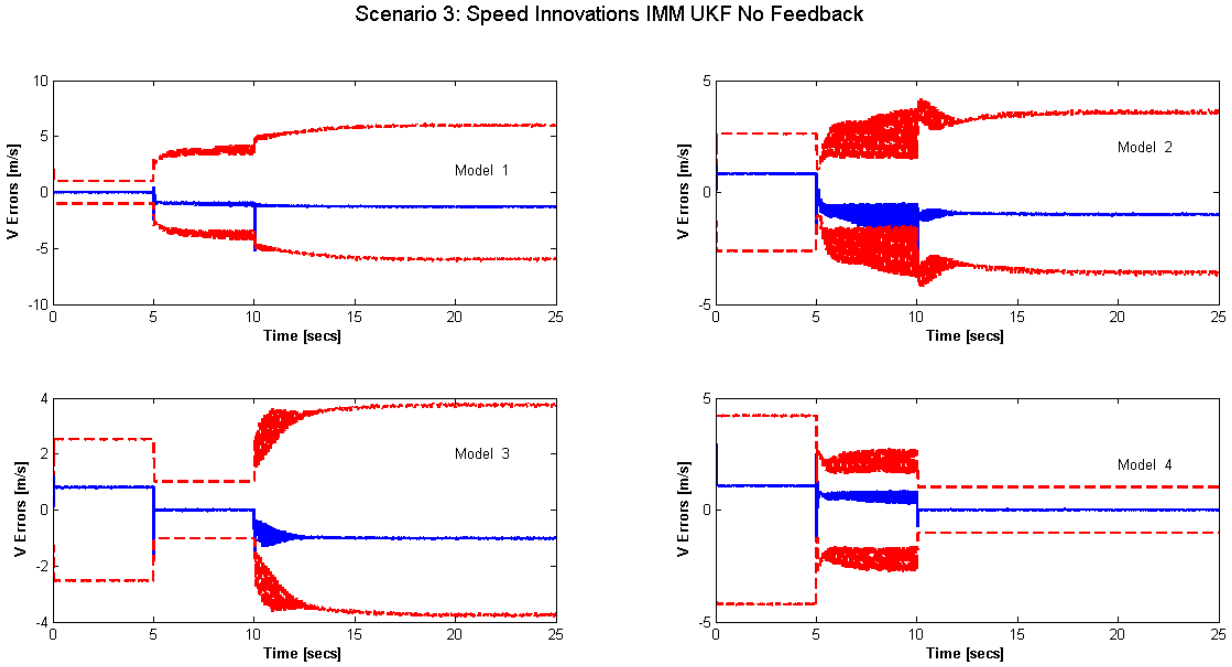

Figure 4.43: Scenario 3 - IMM UKF Speed Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

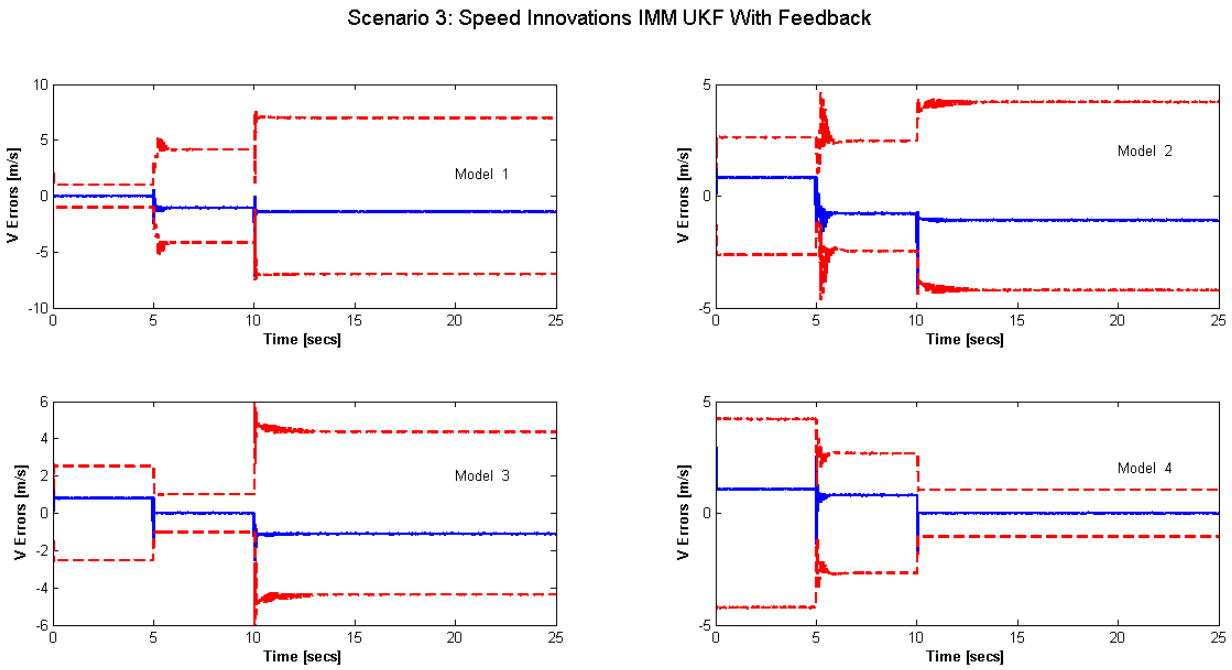

Figure 4.44: Scenario 3 - IMM UKF Speed Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

4.5.3.2 Wheel Radius Estimates

The radius estimates produced by the filters, given in figures 4.45, 4.46, 4.47 and 4.48, show that the IMM filters produce the best estimates of the radii. The UKF performs slightly better than the EKF, and turning feedback on results in a faster settling time.

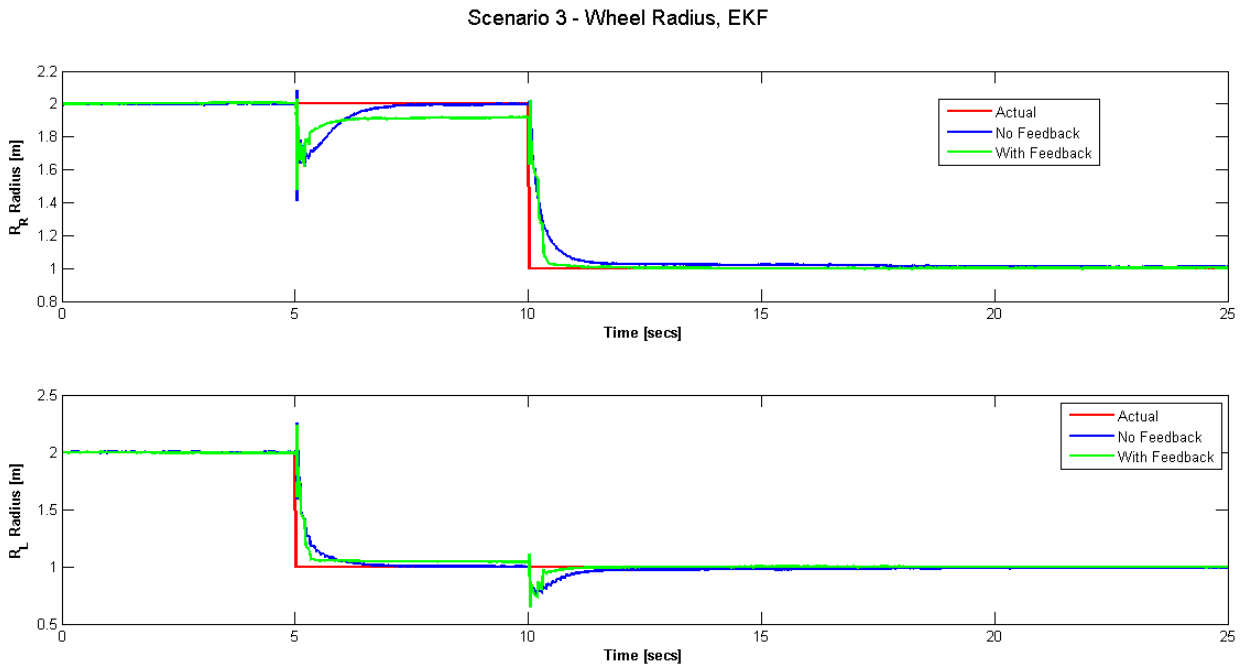

Figure 4.45: Scenario 3 - EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

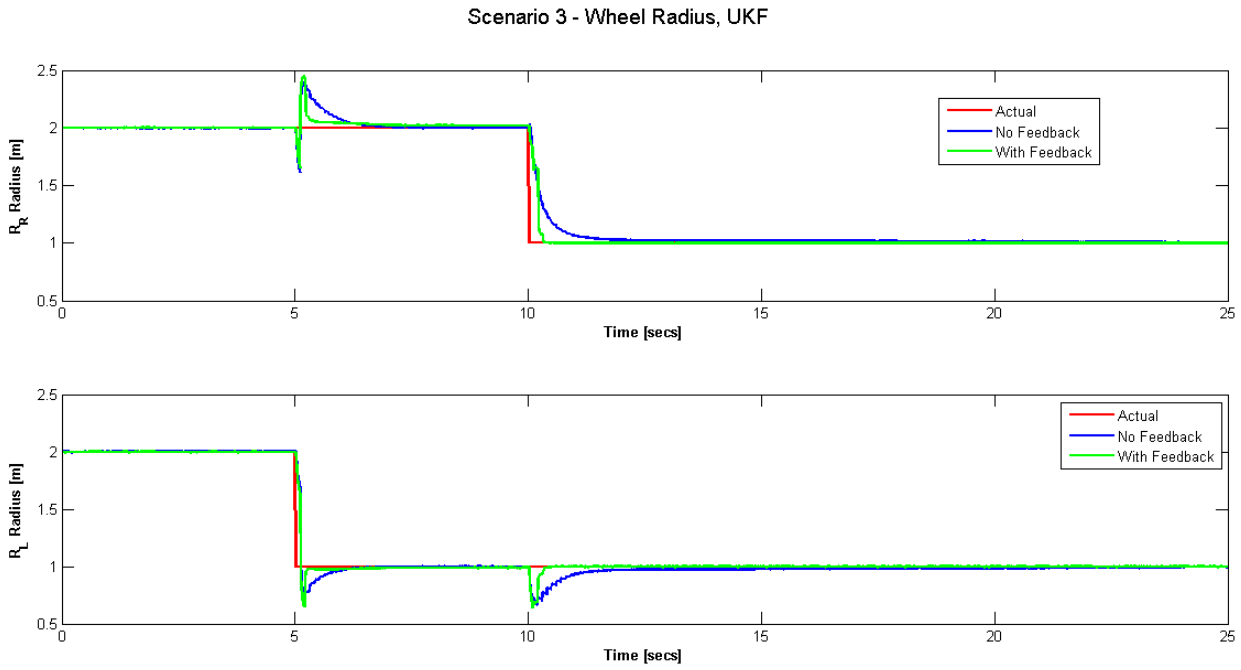

Figure 4.46: Scenario 3 - UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

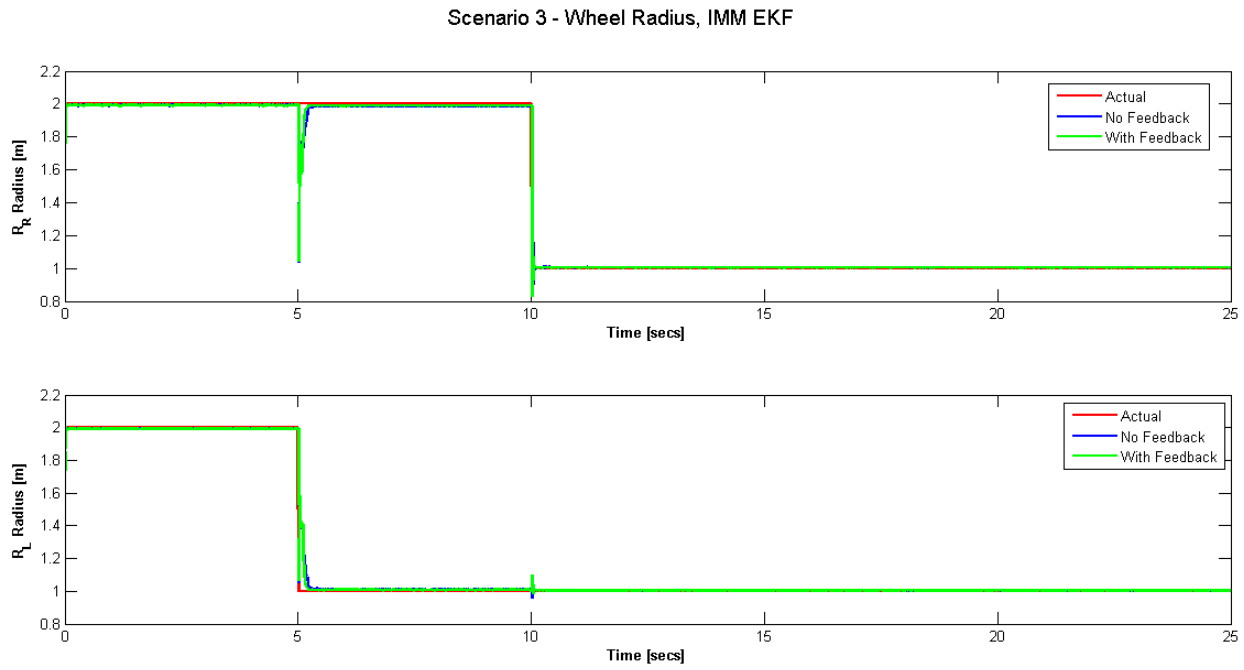

Figure 4.47: Scenario 3 - IMM EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

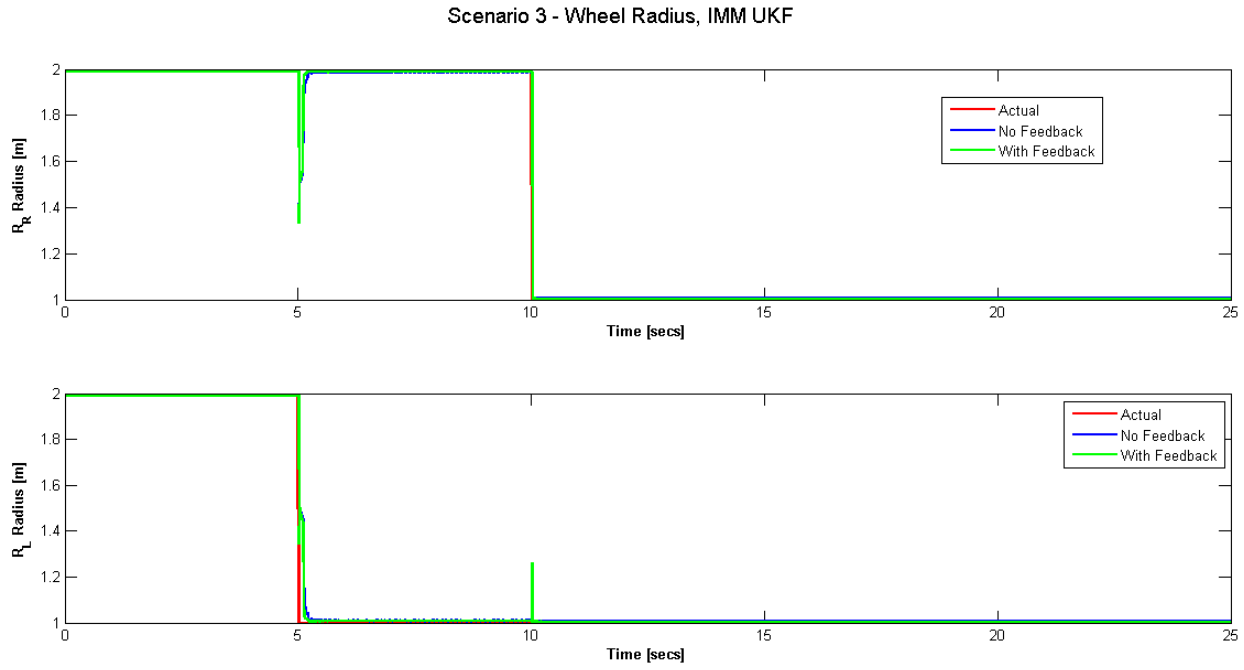

Figure 4.48: Scenario 3 - IMM UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

4.5.3.3 Wheel Speeds

Figure 4.49 presents the angular rates achieved by the robot as a result of EKF estimates which show that once a fault occurs that particular wheel is required to spin faster in order to

compensate for the loss in radius. The angular rates given in figure 4.50 show that estimates used by the UKF to reconfigure the controller resulted in operation at the constraints. Both IMM filters displayed similar behaviour in that once a wheel was punctured it was required to rotate faster to compensate for the loss in radius.

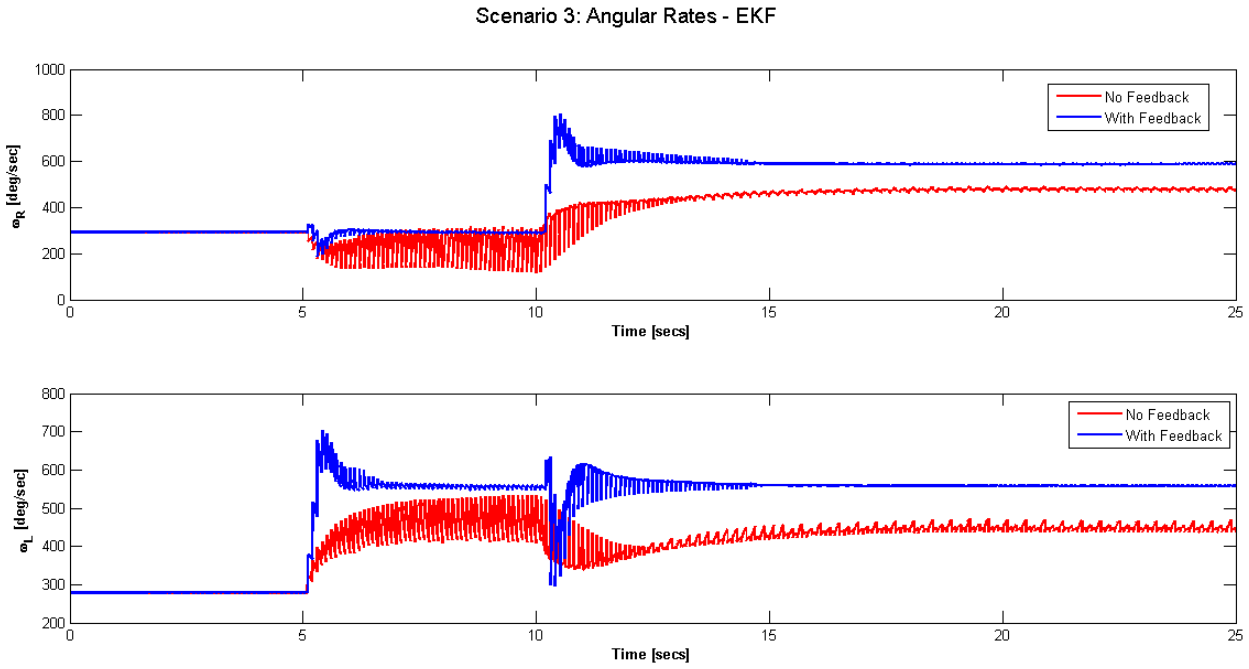

Figure 4.49: Scenario 3 - EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

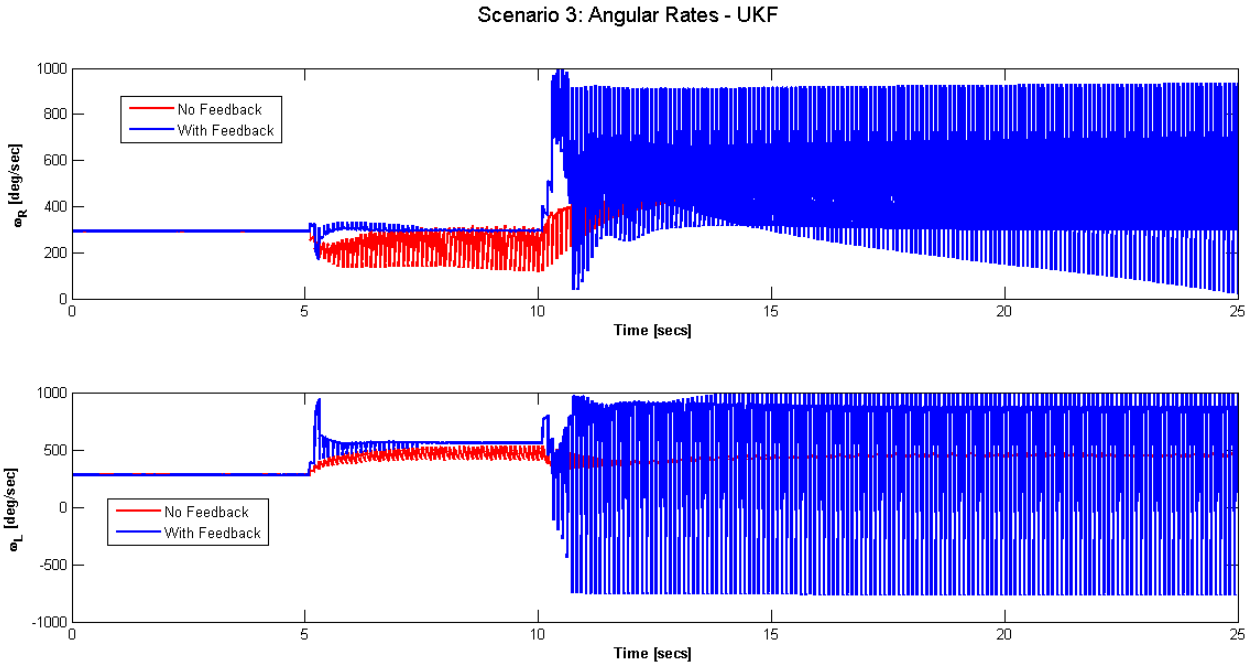

Figure 4.50: Scenario 3 - UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

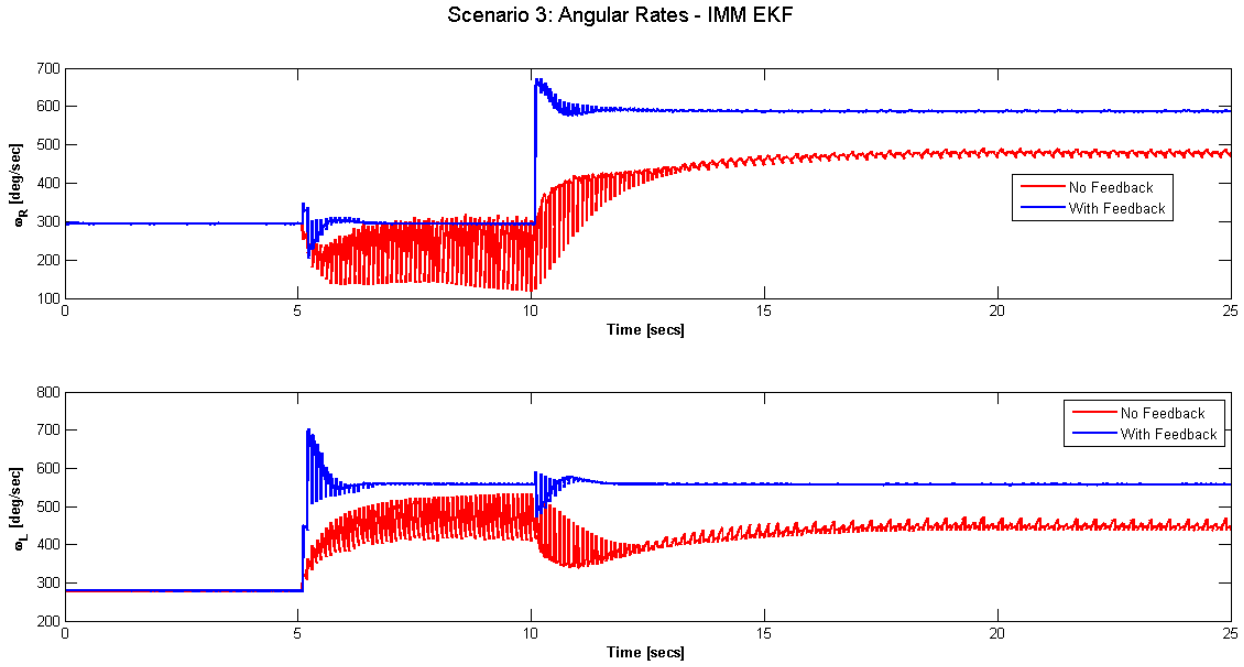

Figure 4.51: Scenario 3 - IMM EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

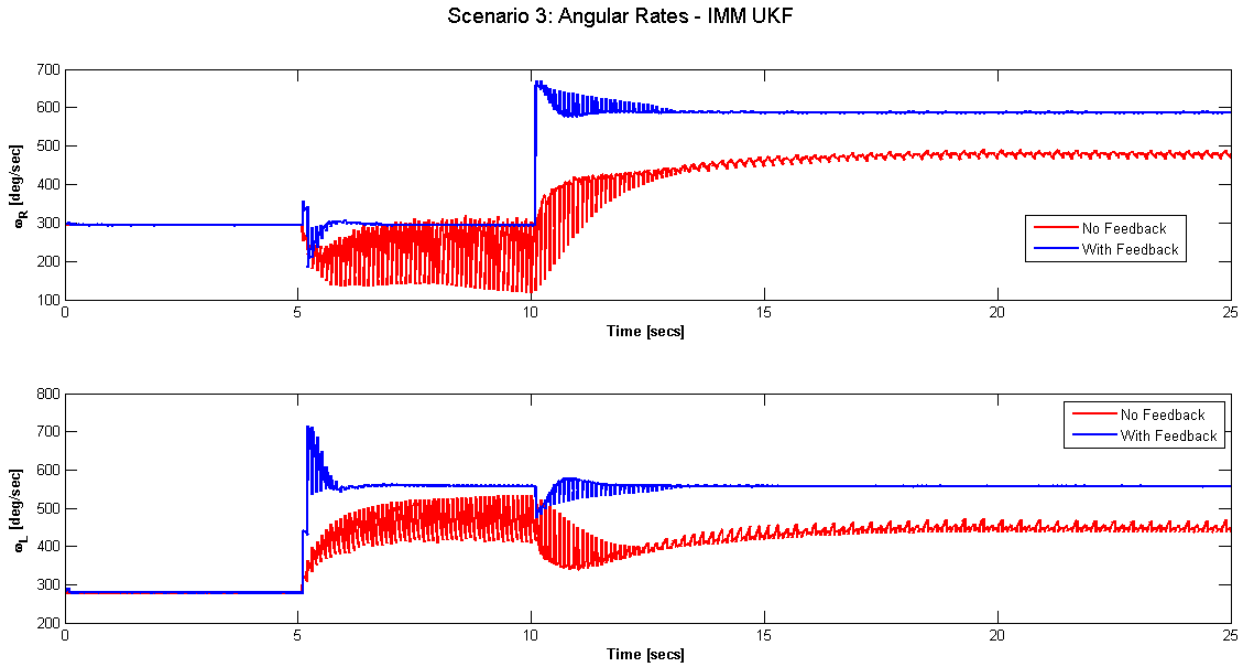

Figure 4.52: Scenario 3 - IMM UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

4.5.3.4 Trajectories

The trajectory produced by each filter for scenario 3 (figures 4.53, 4.54, 4.55 and 4.56) show that without feedback it is impossible to maintain the robot on the path. Reconfiguring the

controller on the other hand with estimates from the filters allowed the robot to easily follow the reference path. An anomaly occurred with the UKF filter where even turning the feedback on did not result in the robot following the path after the occurrence of the second fault. This could possibly be the result of poor tuning of the filter.

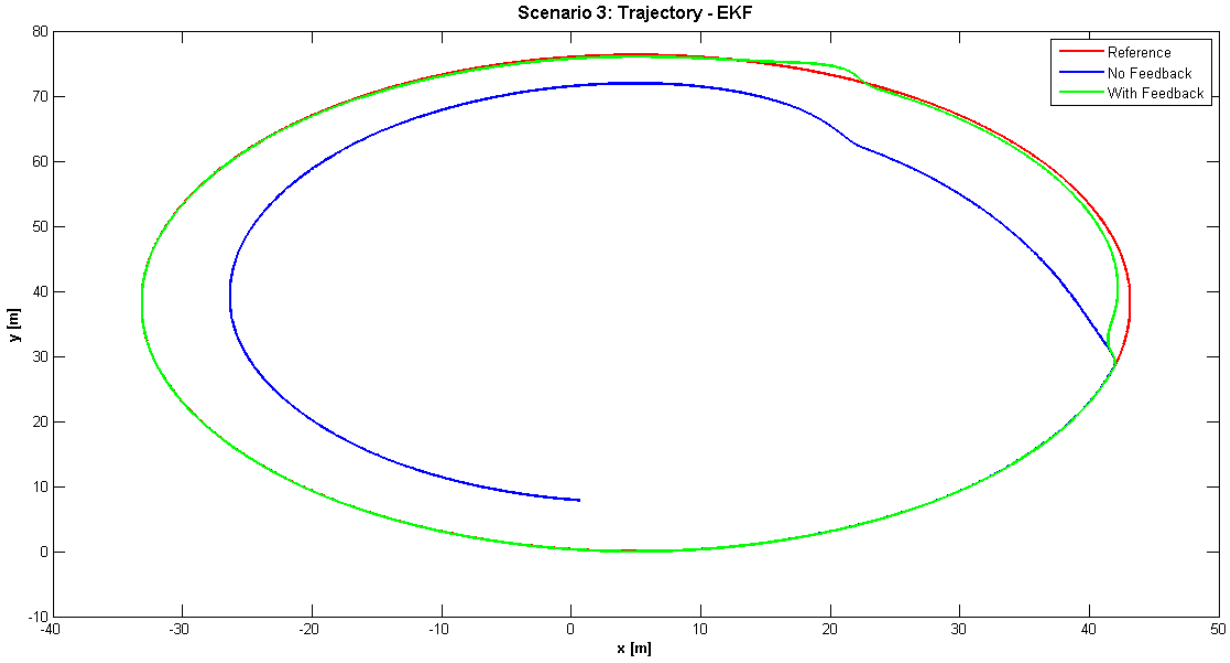

Figure 4.53: Scenario 3 - EKF Trajectory

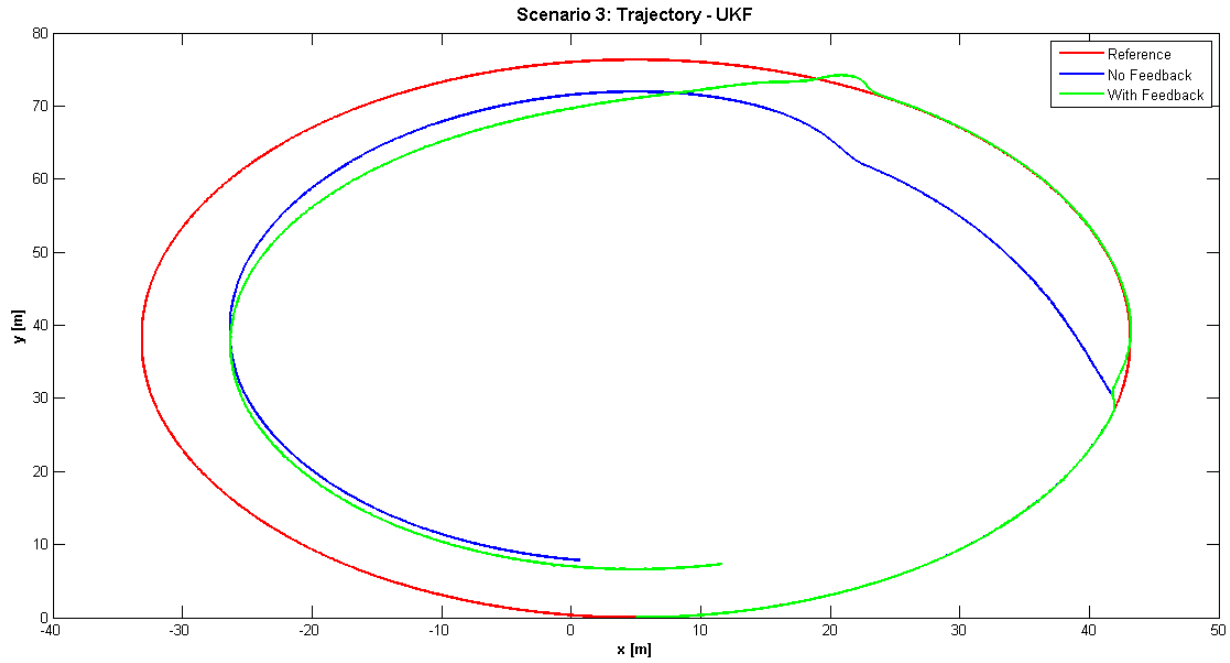

Figure 4.54: Scenario 3 - UKF Trajectory

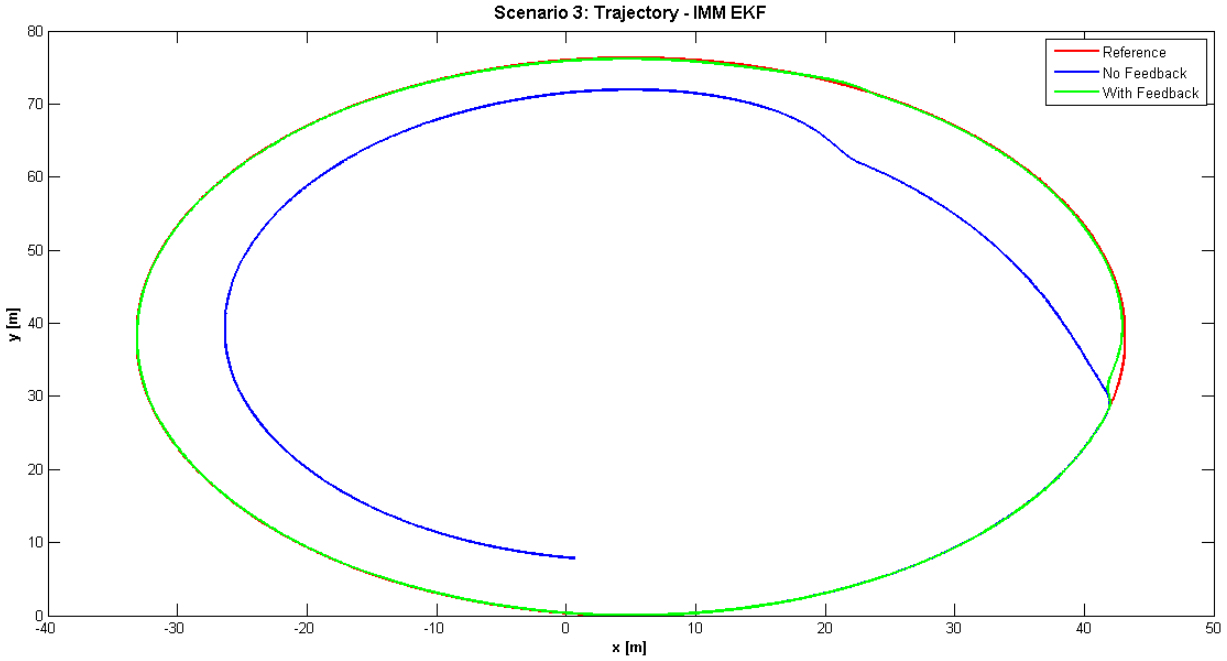

Figure 4.55: Scenario 3 - IMM EKF Trajectory

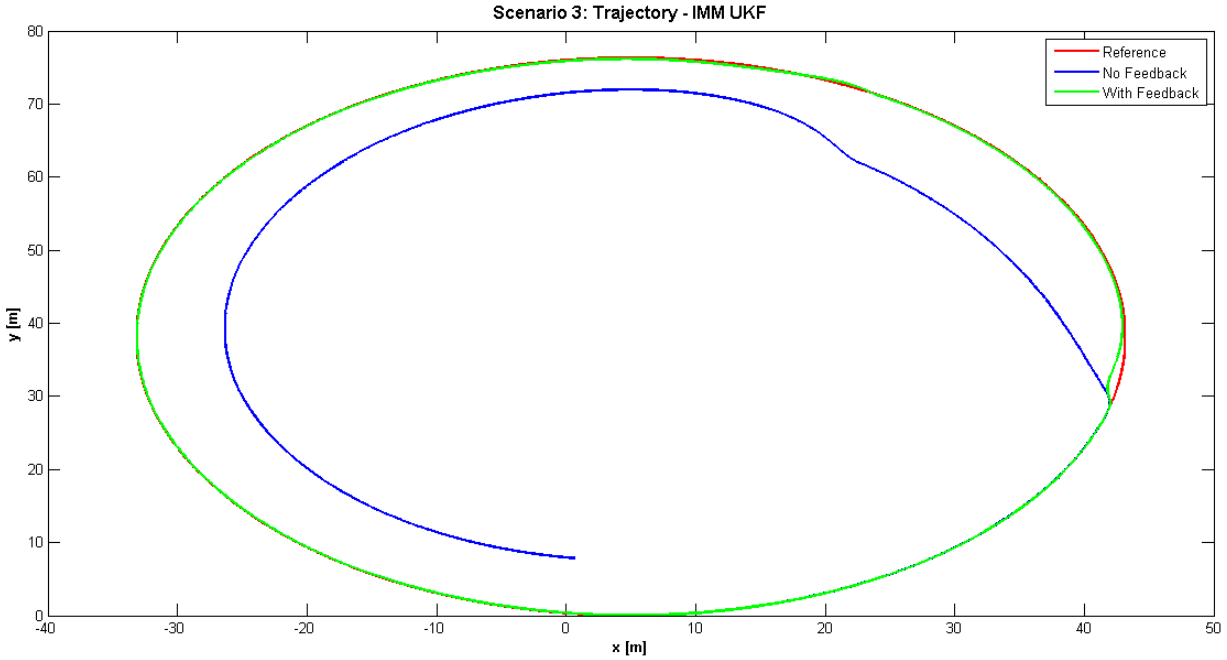

Figure 4.56: Scenario 3 - IMM UKF Trajectory

4.5.4 Scenario 4

4.5.4.1 Speed Errors (Innovations)

Scenario 4 velocity innovations are presented in figures 4.57 and 4.58 for the EKF and UKF respectively. The EKF breaks down 5 seconds after the fault occurs, when feedback is turned on as can be seen by the innovations exceeding the covariance bounds. Without feedback however the innovations remain within the uncertainty bounds. The results from the UKF however are much better as it produces innovations which remain well within the uncertainty bounds with and without feedback. Both of the IMM filters failed, because the fault type of scenario 4 was not modelled as a part of the filter design, i.e. the hypothesis for this type of failure is not accounted for and so no model exists for this failure.

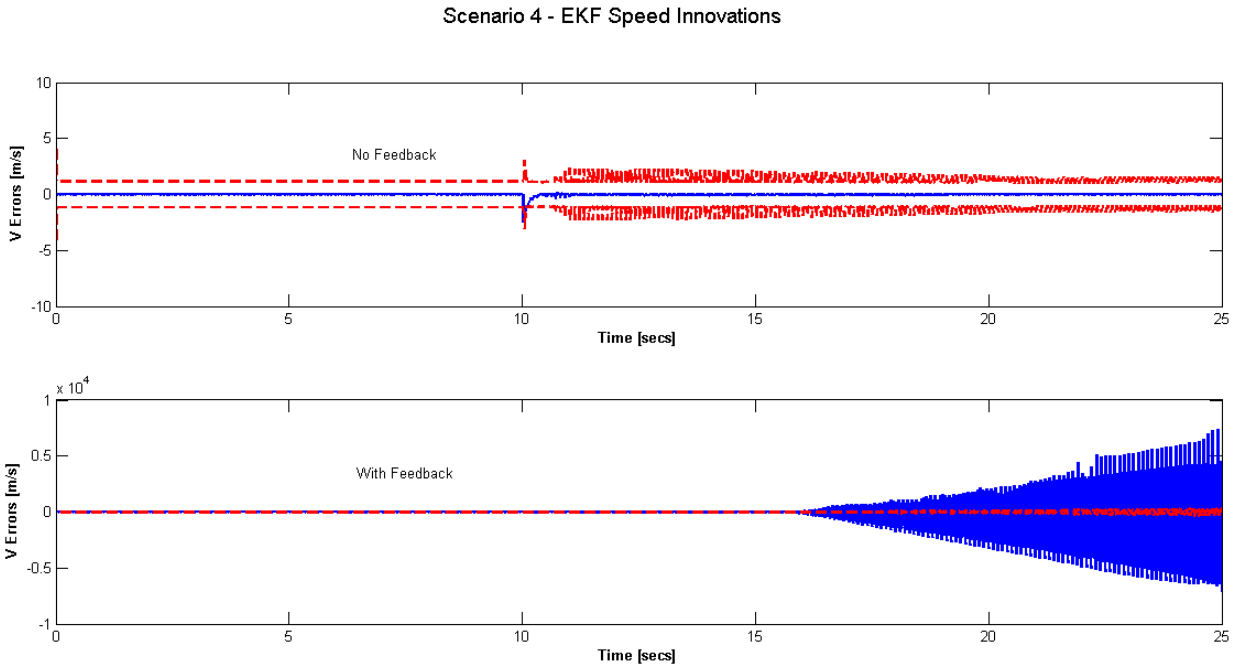

Figure 4.57: Scenario 4 - EKF Velocity Innovations, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

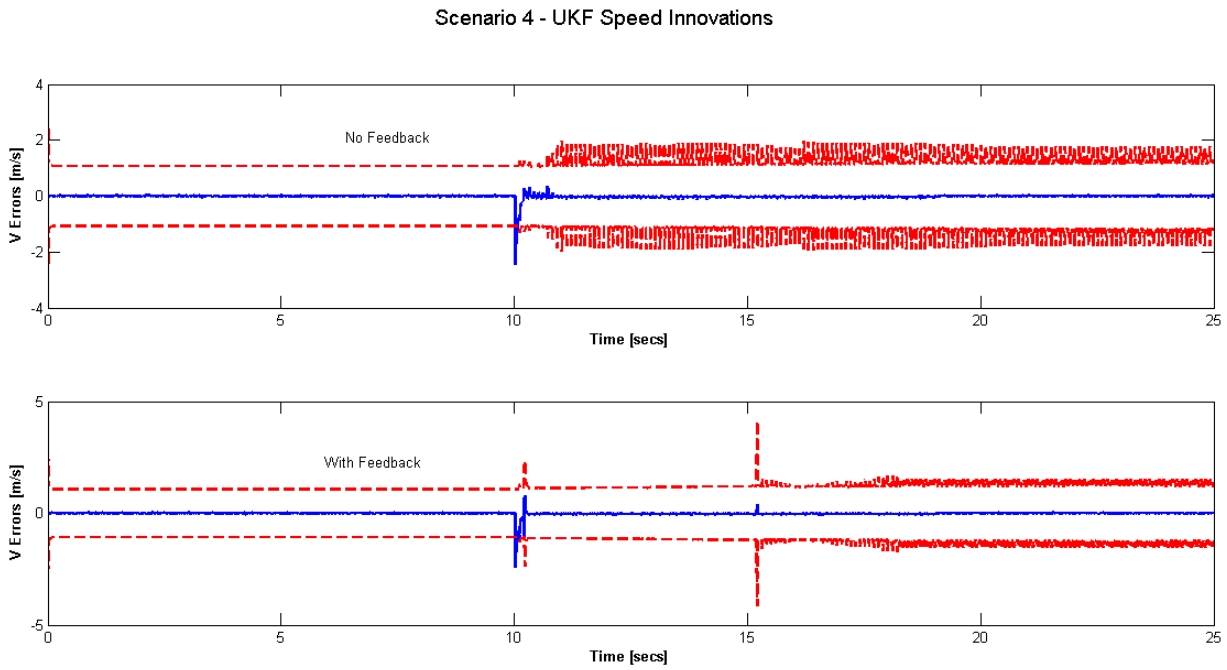

Figure 4.58: Scenario 4 - UKF Velocity Innovations, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

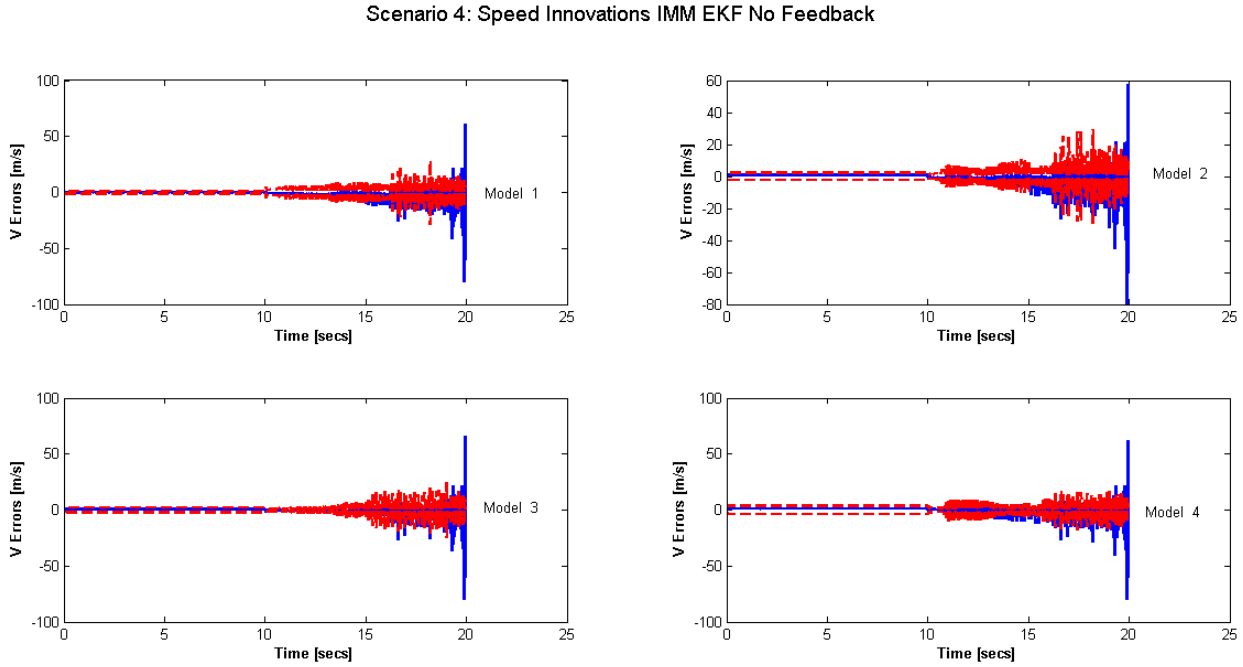

Figure 4.59: Scenario 4 - IMM EKF Velocity Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

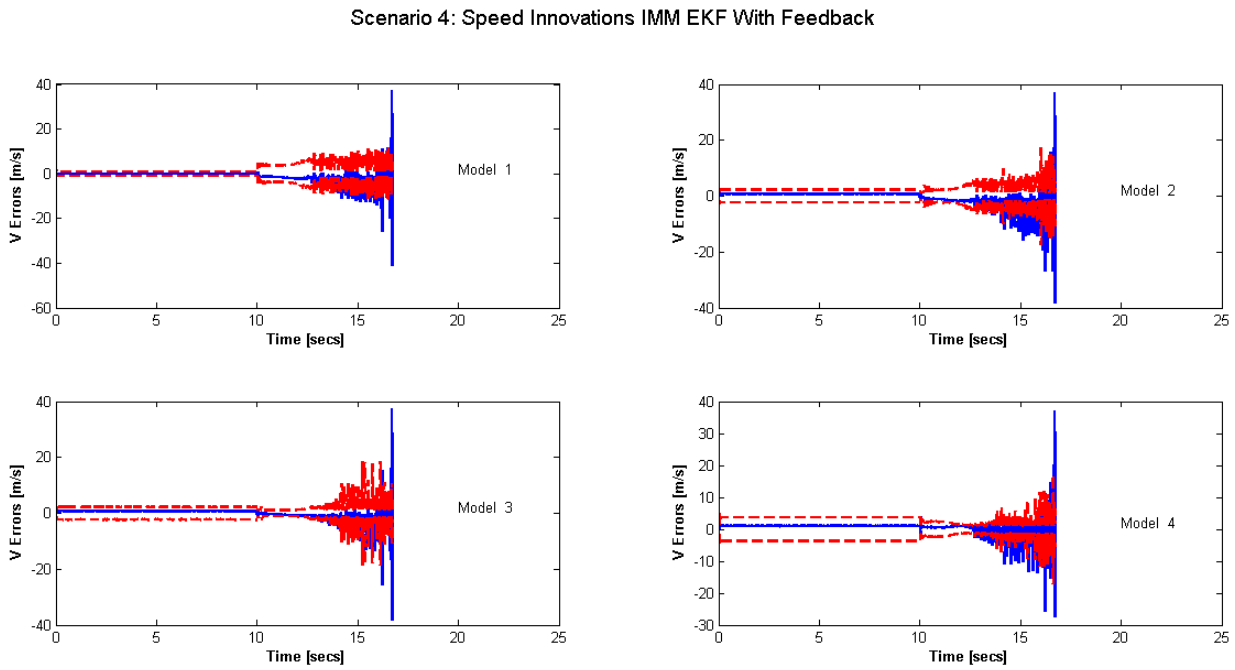

Figure 4.60: Scenario 4 - IMM EKF Velocity Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

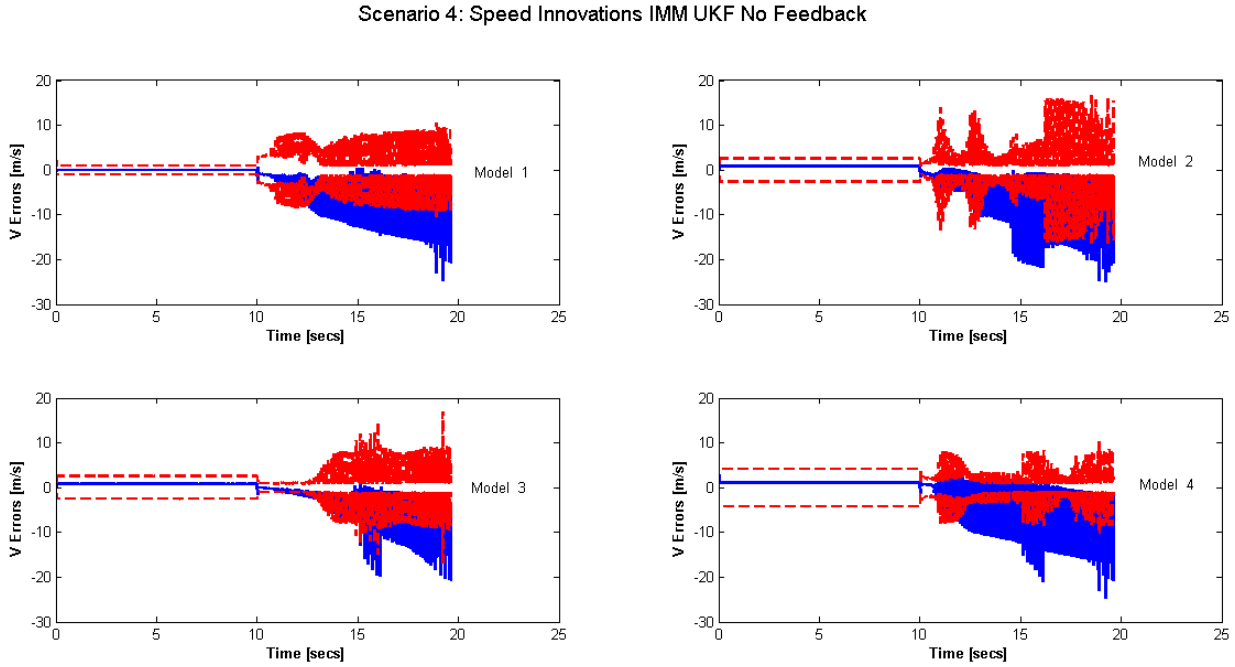

Figure 4.61: Scenario 4 - IMM UKF Velocity Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

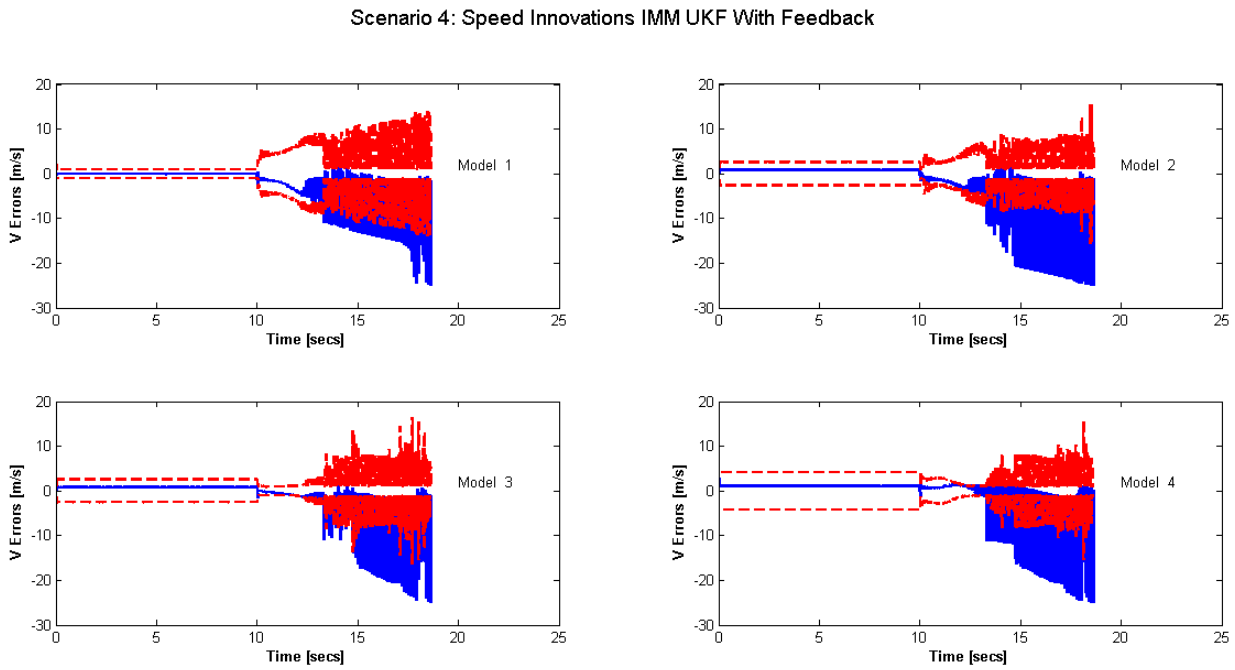

Figure 4.62: Scenario 4 - IMM UKF Velocity Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

4.5.4.2 Wheel Radius Estimates

The wheel radius estimates are given in figures 4.63, 4.64, 4.65 and 4.66. The results show that without a hypothesis on the IMM filters the UKF was the only filter able to produce the correct estimates of the wheel radii. Figure 4.64 clearly indicates that reconfiguring the controller resulted in a faster convergence to the correct estimate.

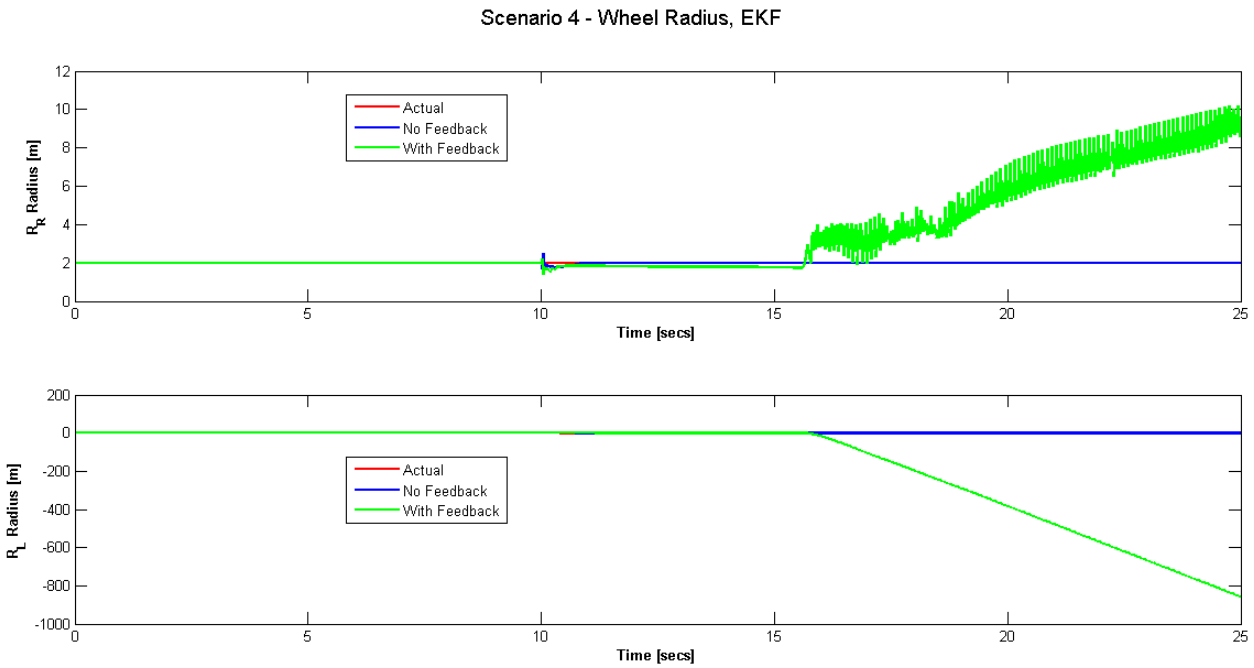

Figure 4.63: Scenario 4 - EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

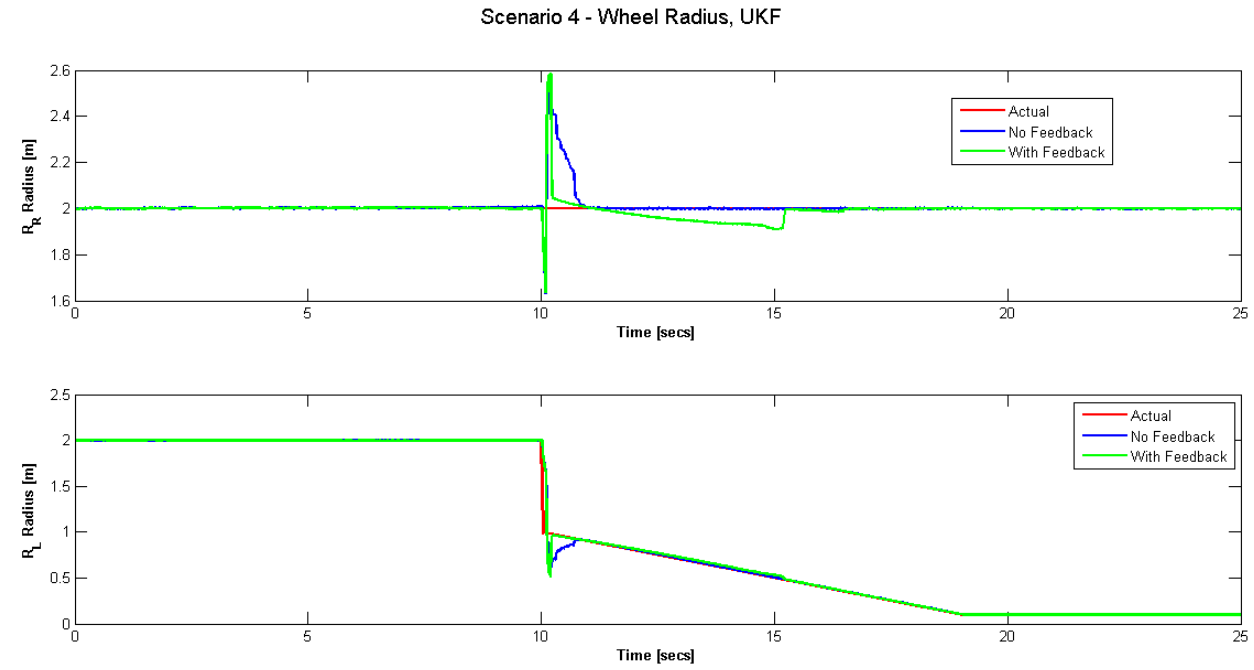

Figure 4.64: Scenario 4 - UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

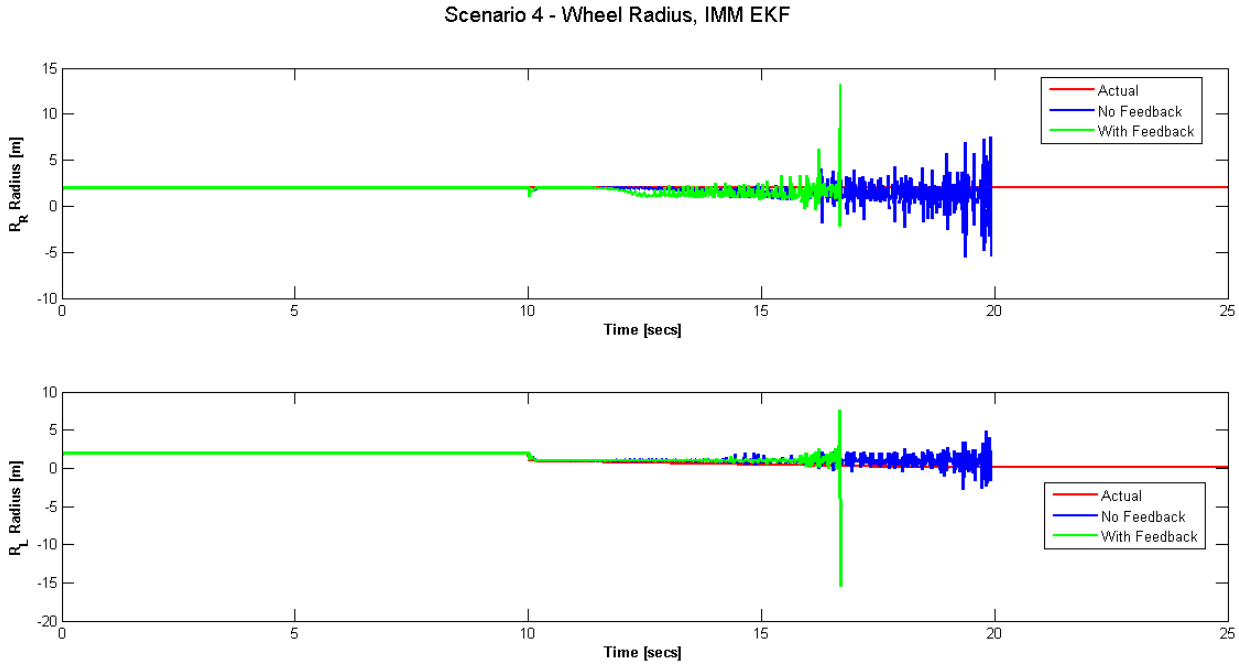

Figure 4.65: Scenario 4 - IMM EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

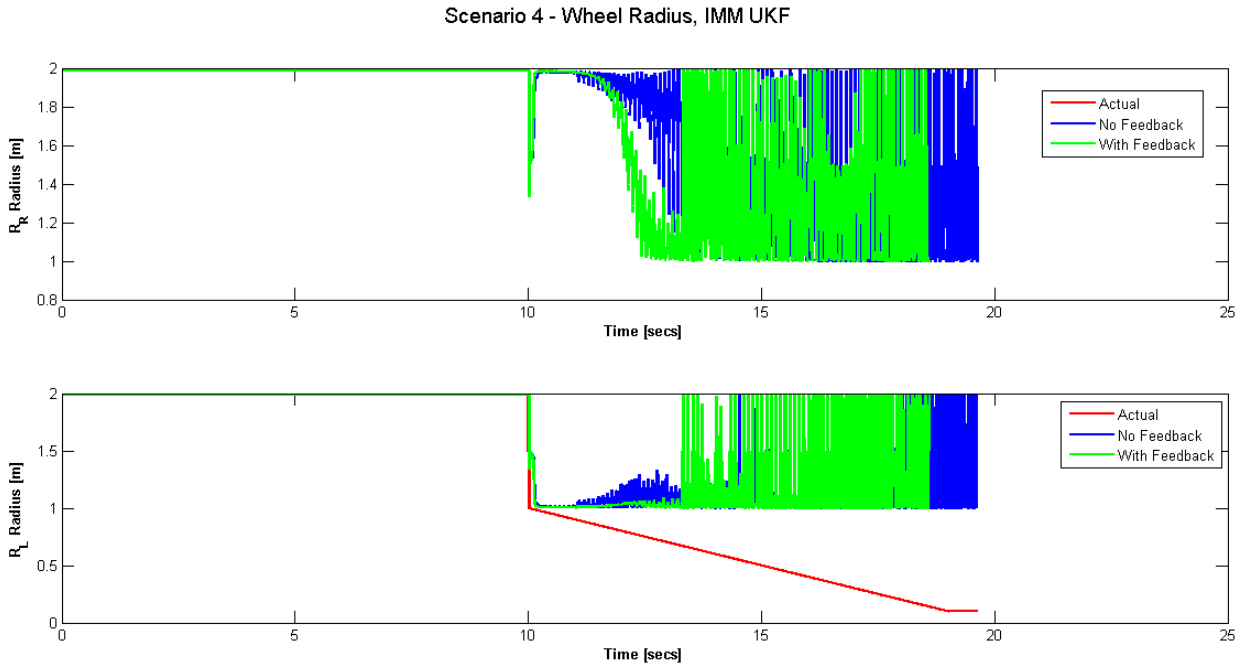

Figure 4.66: Scenario 4 - IMM UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

4.5.4.3 Wheel Speeds

Plots of wheel speed are presented in figures 4.67, 4.68, 4.69 and 4.70. The results produced by all four filters with and without feedback show that with this type of fault both wheels are

required to work at the constraints.

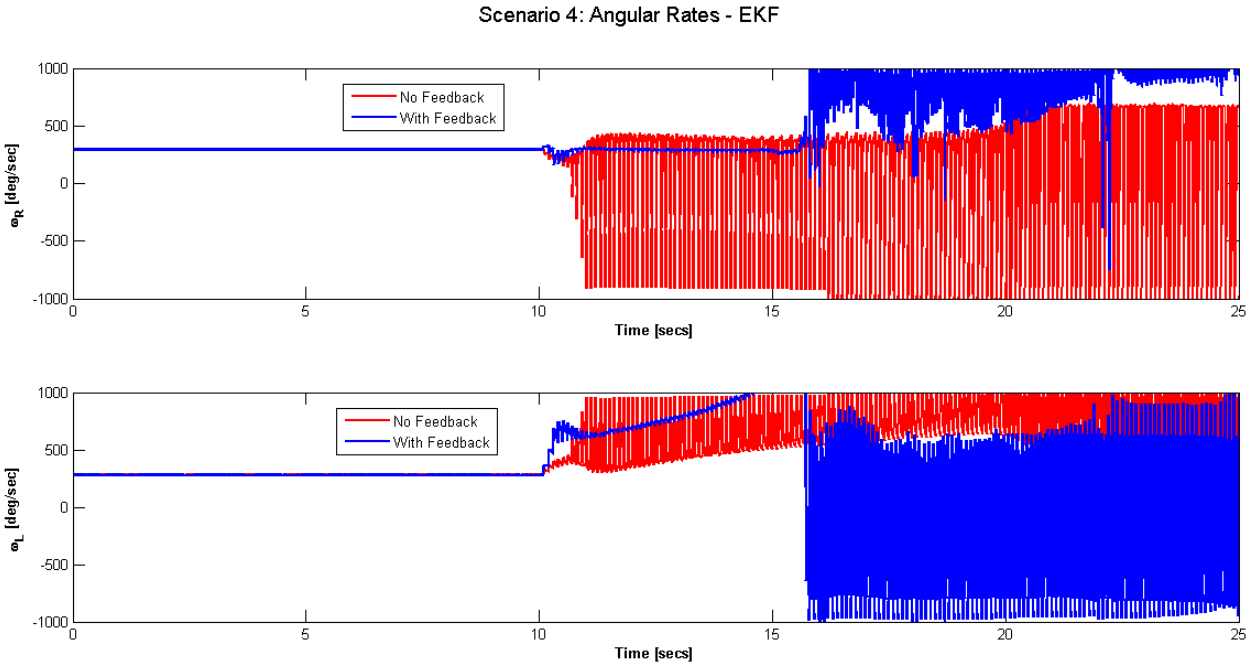

Figure 4.67: Scenario 4 - EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

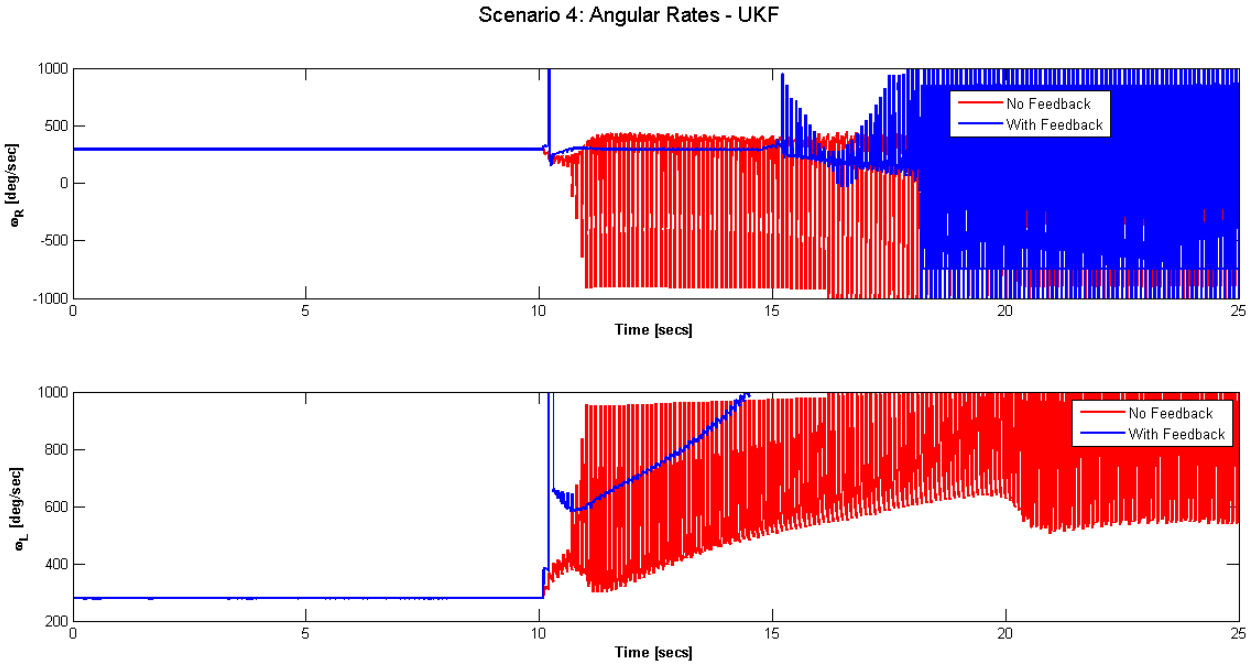

Figure 4.68: Scenario 4 - UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

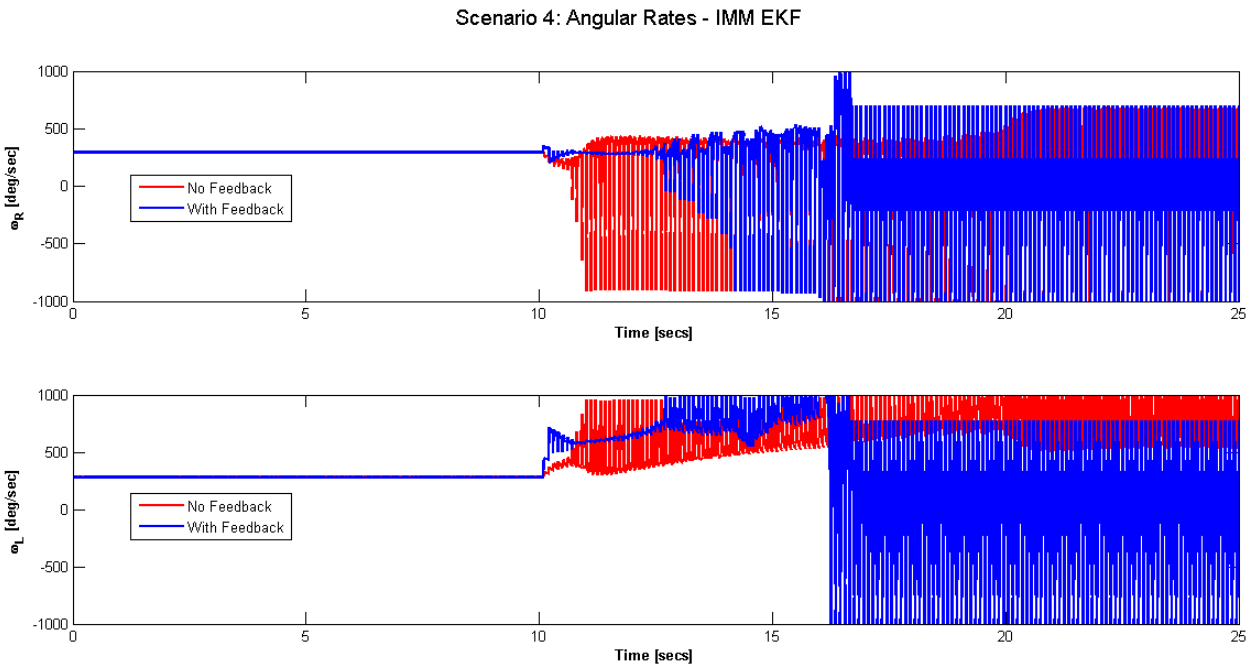

Figure 4.69: Scenario 4 - IMM EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

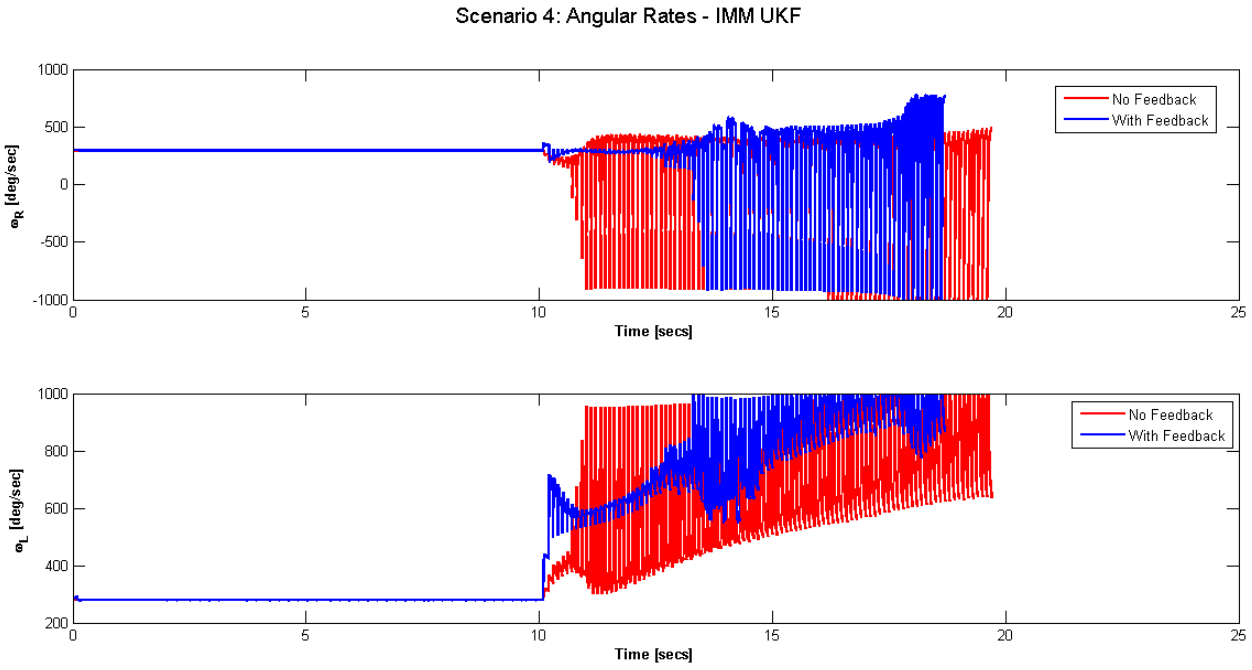

Figure 4.70: Scenario 4 - IMM UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

4.5.4.4 Trajectories

Figures 4.71, 4.72, 4.73 and 4.74 display the robot trajectory as a result of the different filter information. None of the filters show full compliance with the reference trajectory, however the

UKF was able to maintain the robot on the path the longest.

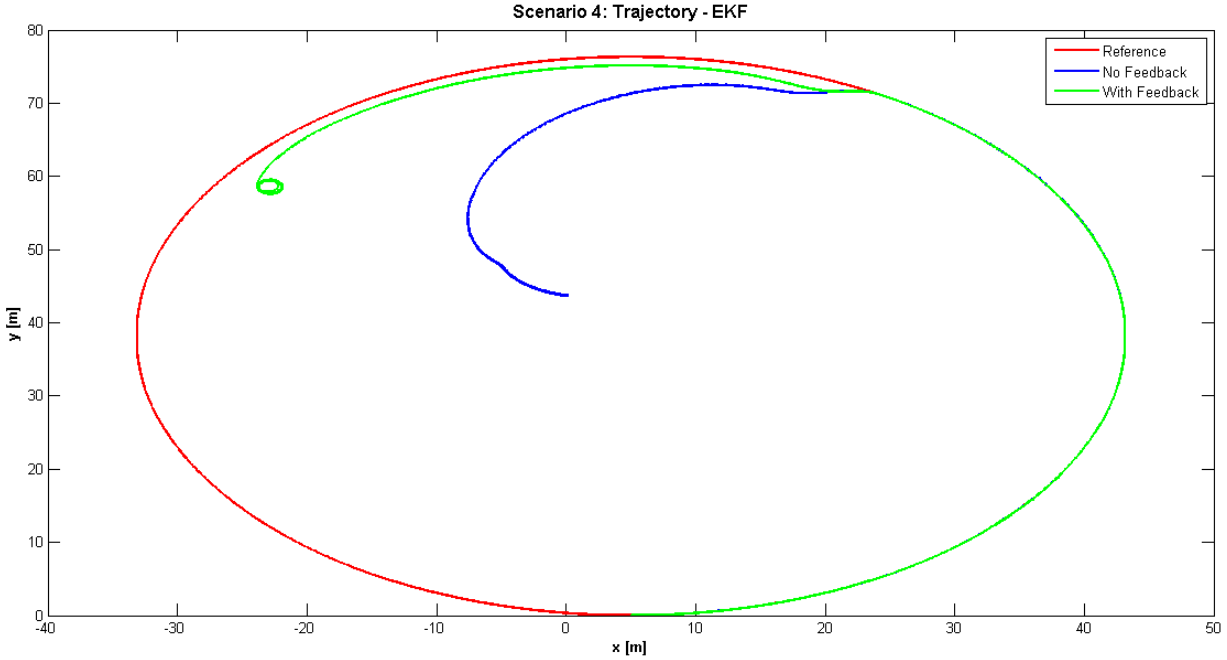

Figure 4.71: Scenario 4 - EKF Trajectory

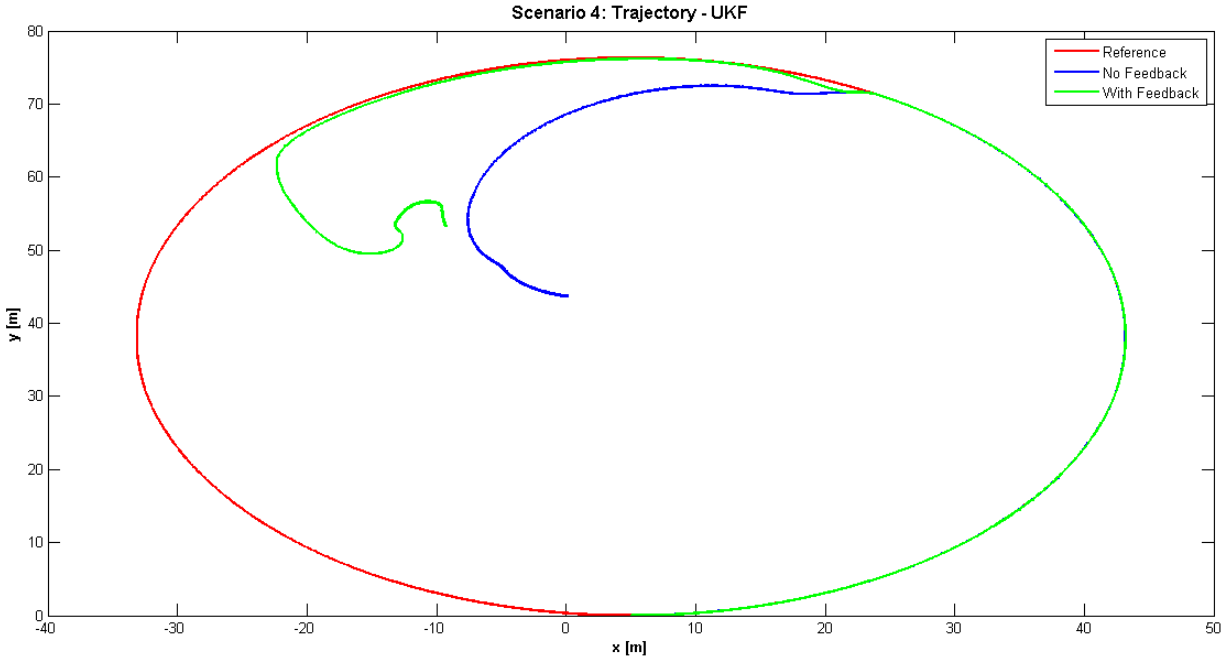

Figure 4.72: Scenario 4 - UKF Trajectory

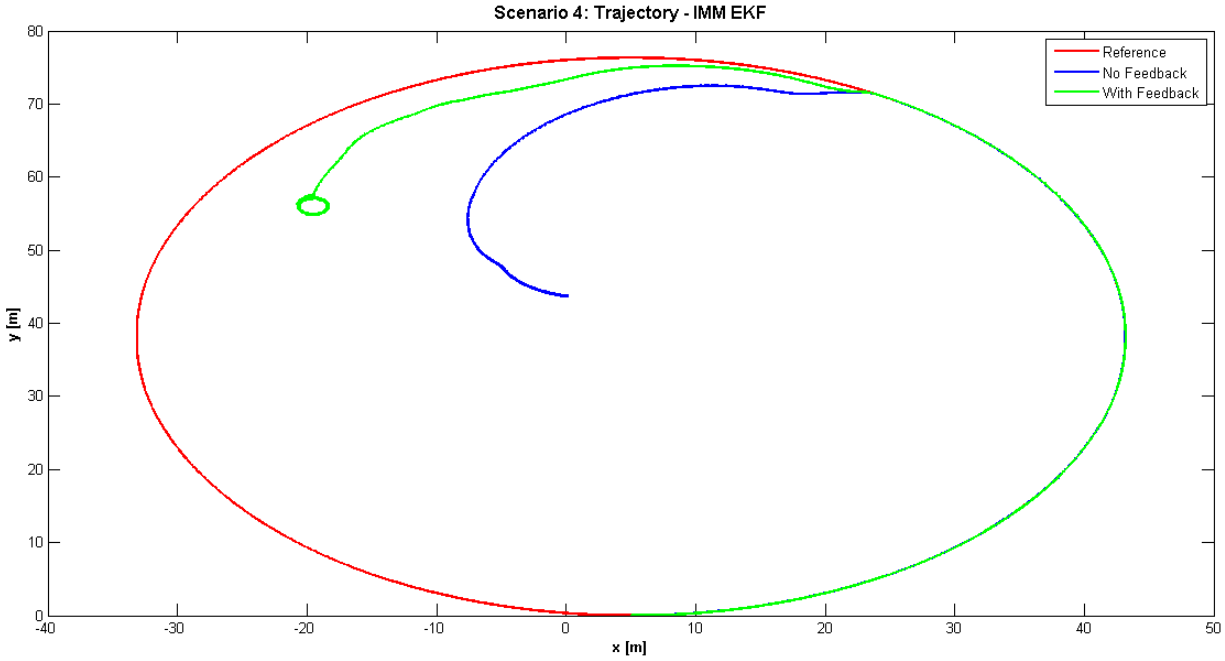

Figure 4.73: Scenario 4 - IMM EKF Trajectory

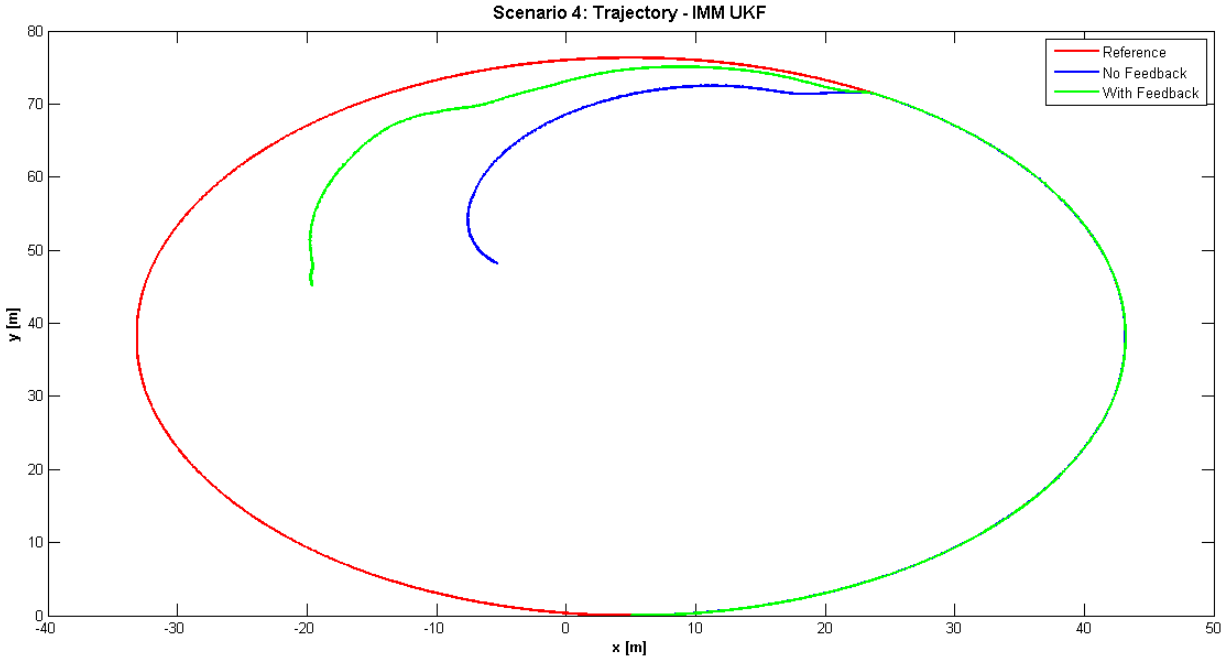

Figure 4.74: Scenario 4 - IMM UKF Trajectory

4.5.4.5 Filter Re-Design

This section looks at the behaviour of the IMM filters by re-designing the filters to accommodate the fault type covered by scenario 4. The filters were modified by adding a fifth model, which hypothesises the left wheel puncturing in the manner described by scenario 4. The five different models are:

Model 1: No Fault case

Model 2: 50% right wheel deflation, left wheel no fault

Model 3: Right wheel no fault, 50% left wheel deflation

Model 4: 50% right wheel deflation, 50% left wheel deflation

Model 5: Right wheel no fault, left wheel deflation according to $R_L = 2 - 0.1t$, where t is the current time and once R_L reaches $0.1m$ it remains constant $0.1m$.

The Q and R remain the same and the initial state vectors for each filter and mode are:

$$\mathbf{x}_1(0) = [x_0,\ y_0\ \psi_0\ 2\ 2]^\intercal, \tag{4.66}$$

$$\mathbf{x}_2(0) = [x_0,\ y_0\ \psi_0\ 1\ 2]^\intercal, \tag{4.67}$$

$$\mathbf{x}_3(0) = [x_0,\ y_0\ \psi_0\ 2\ 1]^\intercal, \tag{4.68}$$

$$\mathbf{x}_4(0) = [x_0,\ y_0\ \psi_0\ 1\ 1]^\intercal, \tag{4.69}$$

$$\mathbf{x}_5(0) = [x_0,\ y_0\ \psi_0\ 2\ 2]^\intercal. \tag{4.70}$$

The mixing probabilities or mode probabilities become:

$$\mu = [1/5, 1/5, 1/5, 1/5, 1/5]^\intercal, \tag{4.71}$$

and the mode transition probabilities matrix p is redefined as:

$$p = \begin{bmatrix} 0.96 & 0.01 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.96 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.96 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.96 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.01 & 0.96 \end{bmatrix}. \tag{4.72}$$

The speed innovations for the IMM EKF are presented in figures 4.75 and 4.76 with no feedback and with feedback respectively. As predicted the results show that if a hypothesis is made the IMM performs very well. The EKF as part of IMM is able to predict this type of error which it was unable to do as a single filter. The IMM UKF results are presented in figures 4.77 and 4.78 without feedback and with respectively. The IMM UKF shows a higher level of confidence in its estimates compared to its EKF counterpart as the uncertainty is lower and consistent. Providing feedback in both cases (EKF and UKF IMMs) was shown to increase confidence in the filter estimates.

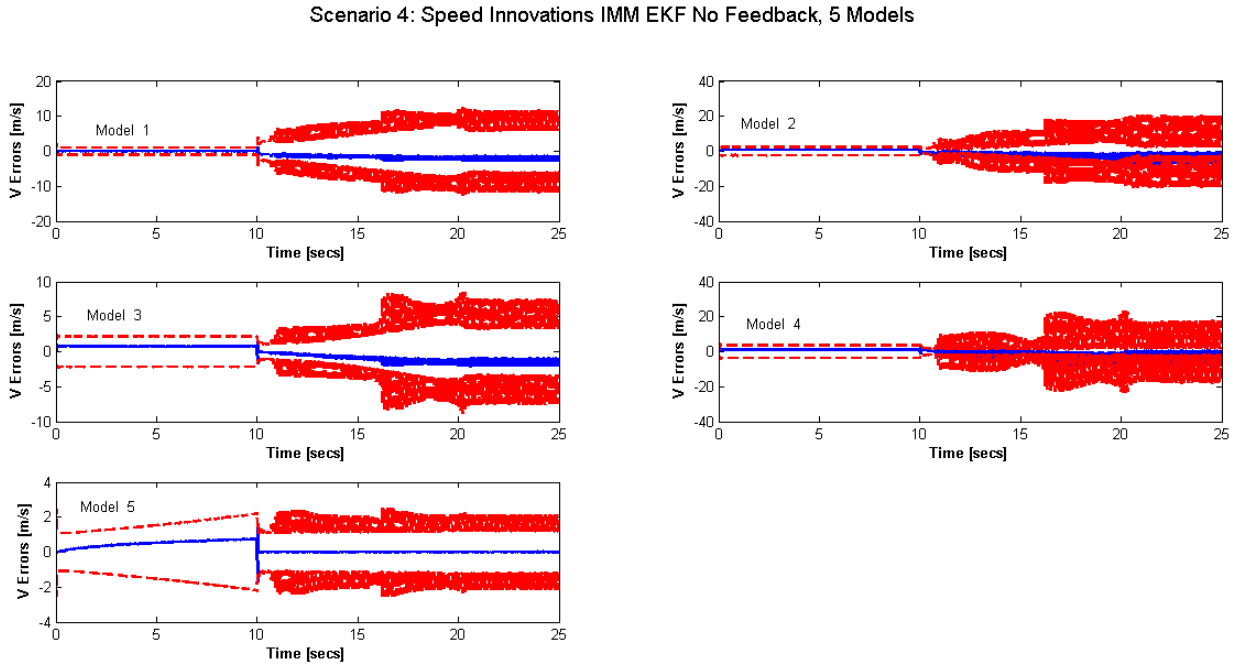

Figure 4.75: Scenario 4 - 5 Model IMM EKF Velocity Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

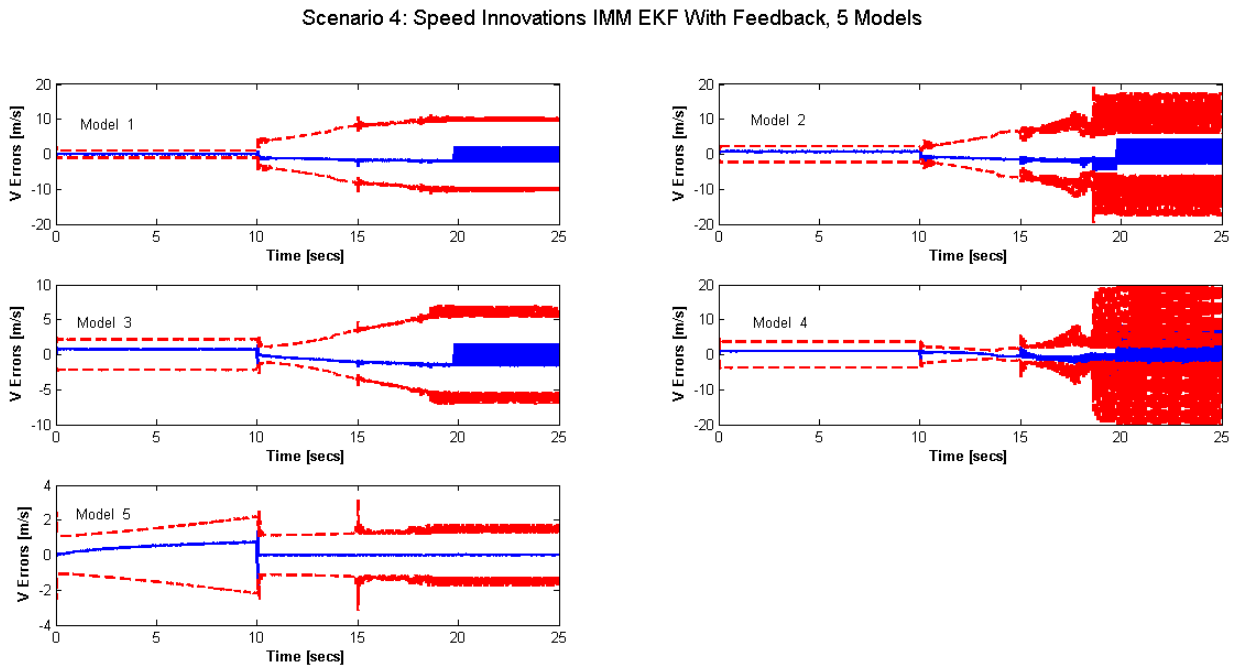

Figure 4.76: Scenario 4 - 5 Model IMM EKF Velocity Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

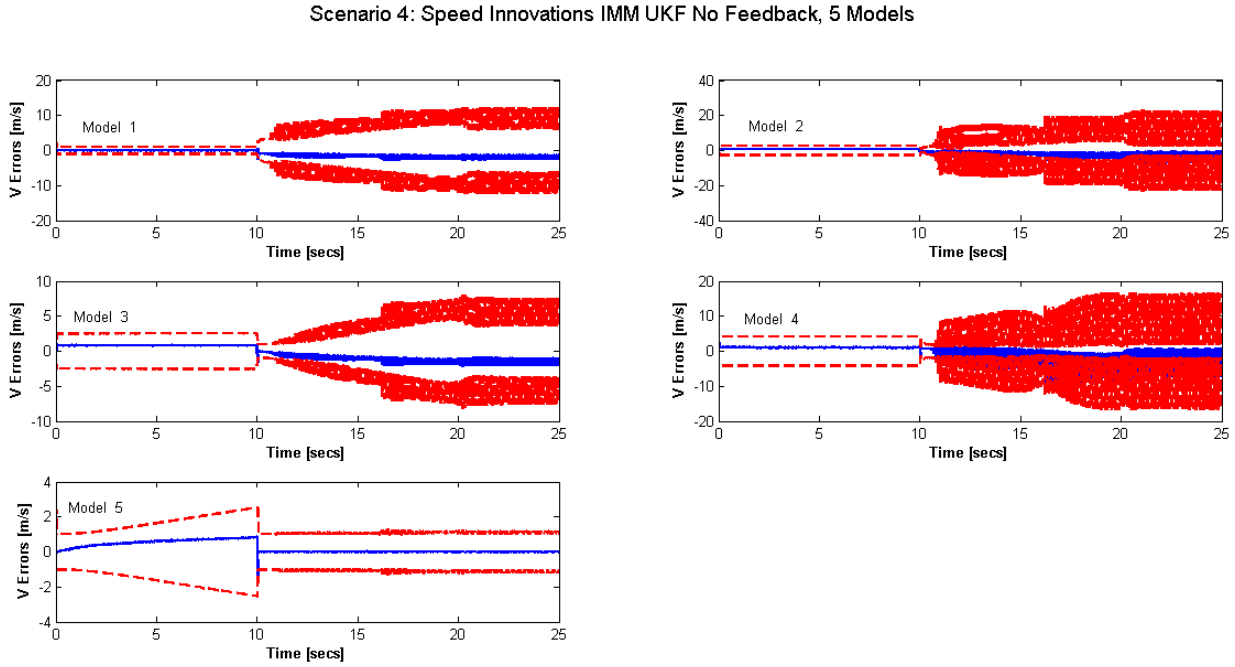

Figure 4.77: Scenario 4 - 5 Model IMM UKF Velocity Innovations No Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

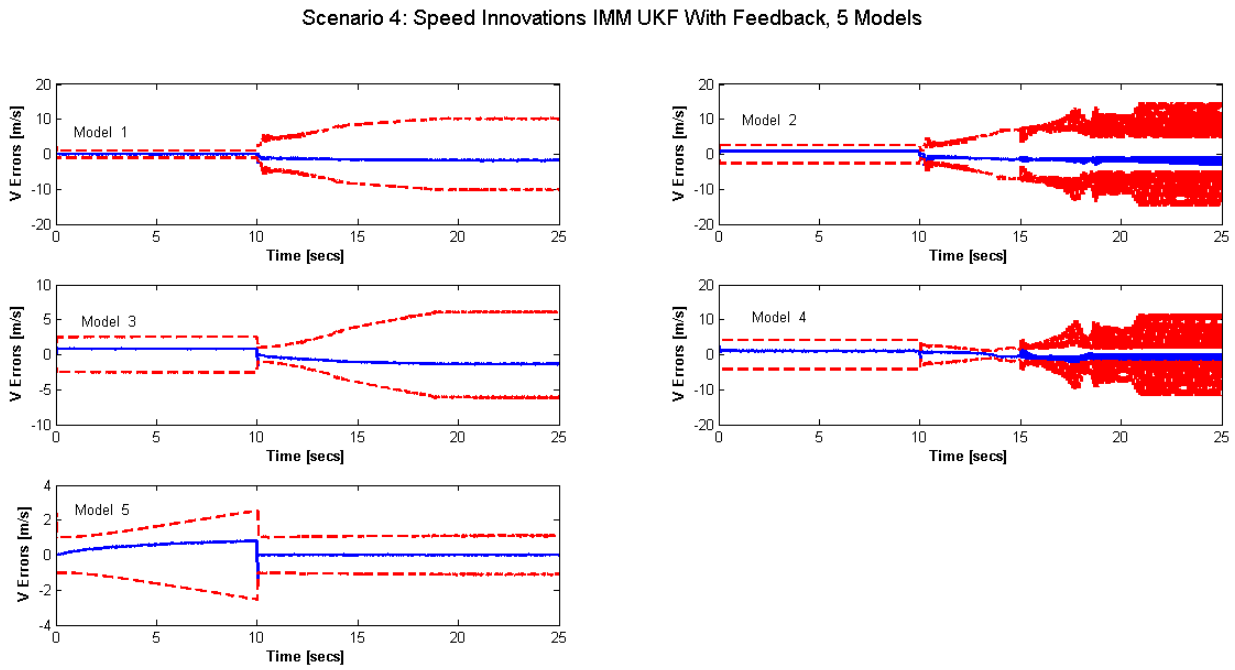

Figure 4.78: Scenario 4 - 5 Model IMM UKF Velocity Innovations With Feedback, 2σ Uncertainty Bounds (dashed lines), Velocity Innovations (solid line)

The estimates of the wheel radii for the 5 mode IMMs are shown in figures 4.79 and 4.80 for the EKF IMM and UKF IMM respectively. The results show that in both cases the filters do an excellent job of making the correct estimatations on the radius of the wheel.

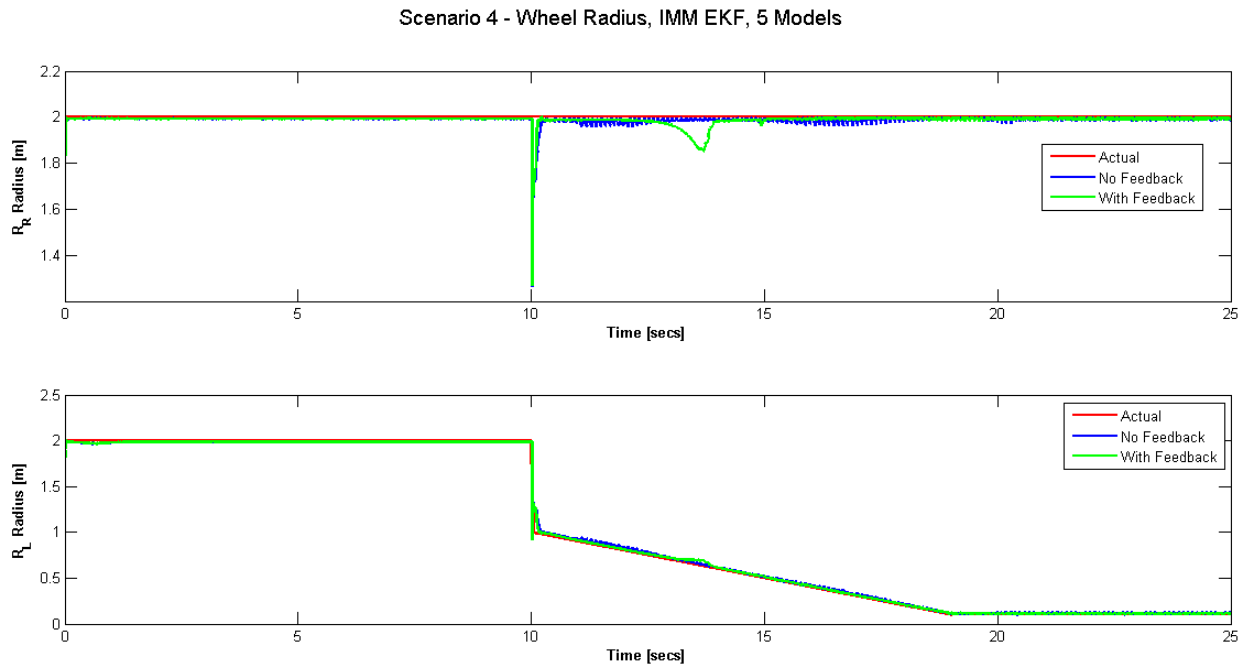

Figure 4.79: Scenario 4 - 5 Model IMM EKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

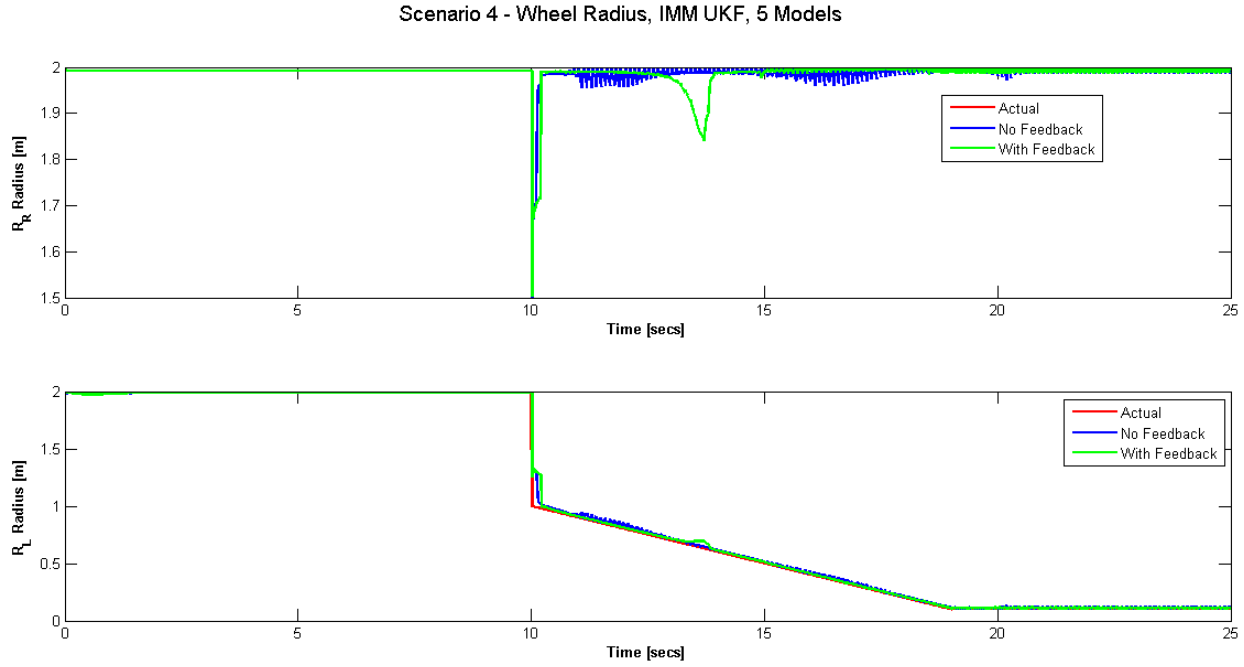

Figure 4.80: Scenario 4 - 5 Model IMM UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

The angular rate plots of the 5 mode IMMs again show that the control inputs are required to

work at the constraints the majority of the time once the fault has occurred. This result is not dependent on the filter type but rather the fault that has occurred makes it impossible for the robot to achieve the desired task while at the same time respecting its constraints.

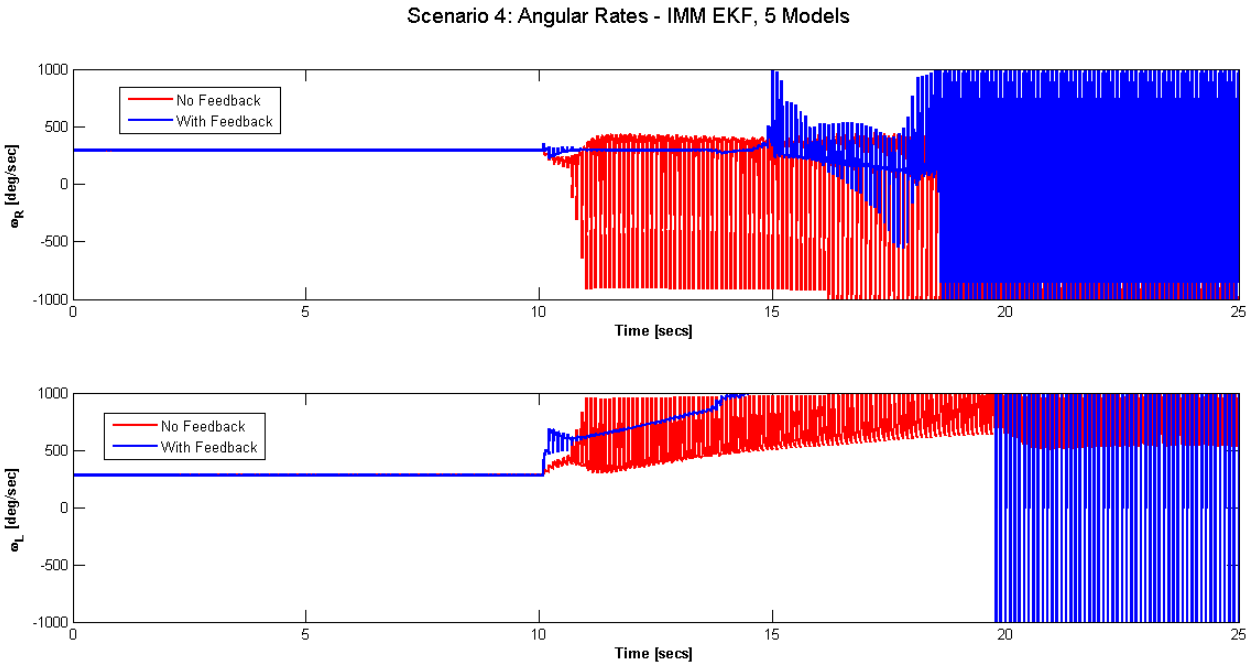

Figure 4.81: Scenario 4 - 5 Model IMM EKF Angular Rates, Right wheel (top), Left wheel (Bottom)

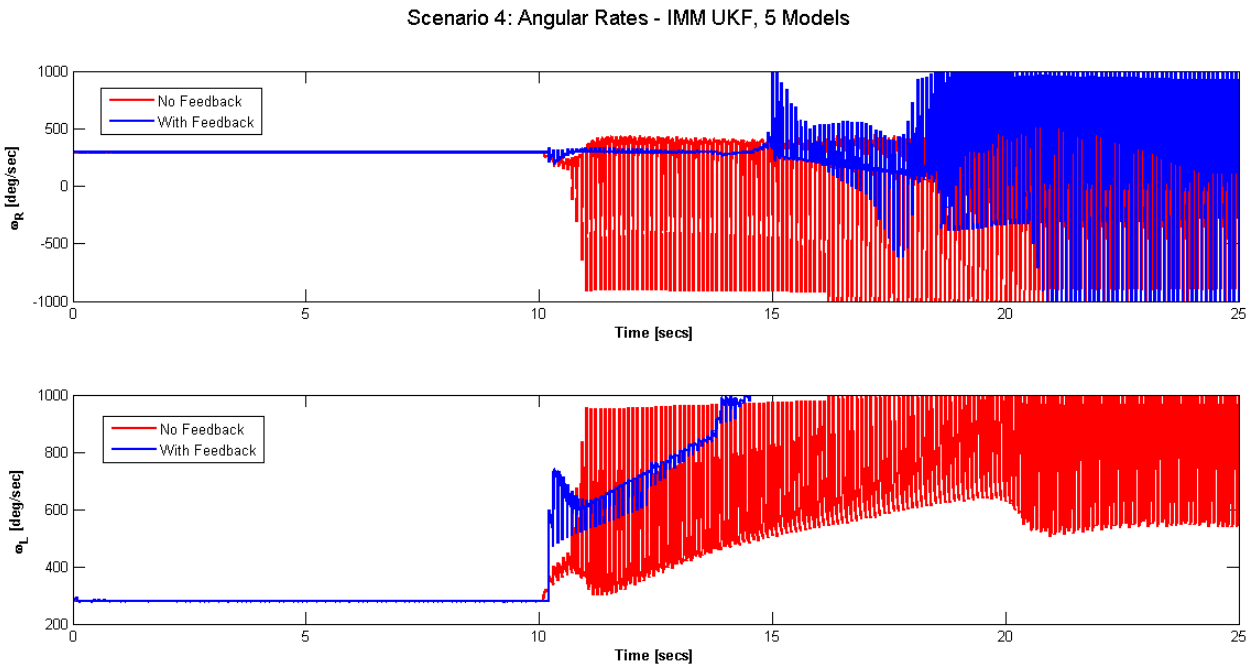

Figure 4.82: Scenario 4 - 5 Model IMM UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

The trajectory plots (figures 4.83 and 4.84) further show that for this type of fault where the wheel radius has almost approached zero, the wheels are unable to maintain the reference. The IMM UKF is again able to keep the robot on the path for a longer time than the IMM EKF.

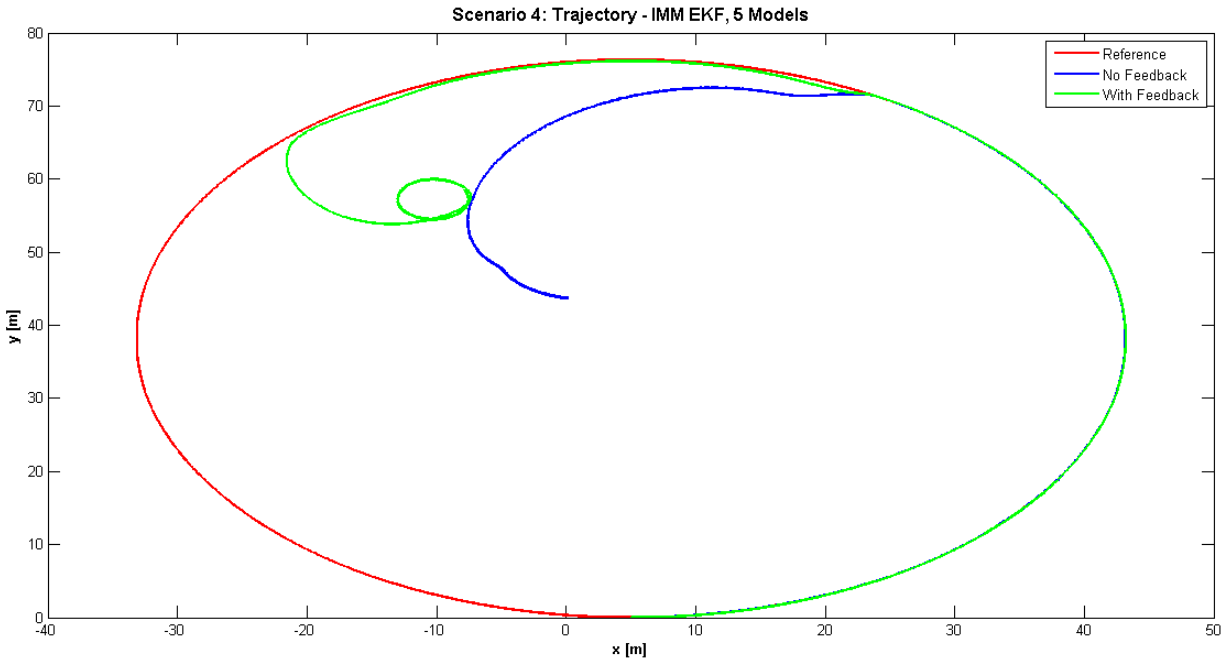

Figure 4.83: Scenario 4 - 5 Model IMM EKF Trajectory

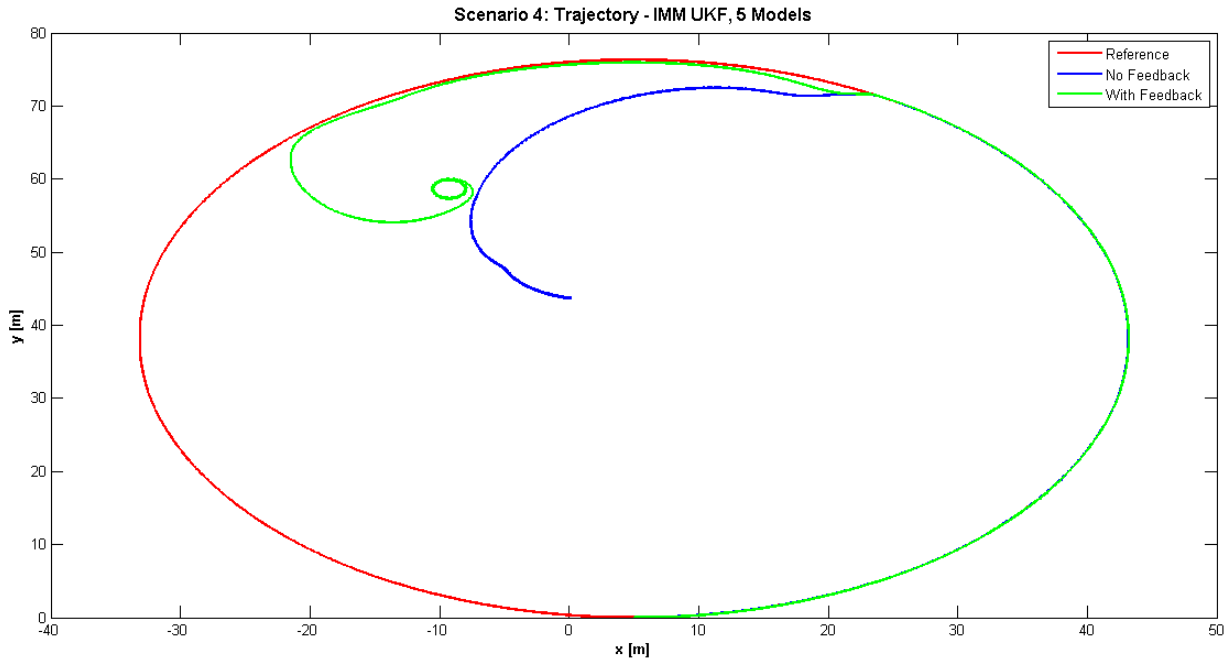

Figure 4.84: Scenario 4 - 5 Model IMM UKF Trajectory

4.5.5 Discussion

The results of this section reveal that all the filters considered exhibit excellent qualities for fault detection and identification. In terms of performance the IMM filter is far superior, however the drawback of this type of filter is that all possible scenarios must be accounted for. The method used in this research illustrated the concept of the IMM, by showing easy adaptability to different situations, and the speed with which it is able to identify and reach the correct estimate. However predicting exactly how a fault will occur (in this case how a tyre will puncture) is impractical. A more practical implementation would have been to develop a number of filters each with different process noises that could adapt to all different situations. The number of filters and the process noise values would have to be determined by trial and error. In either case an IMM performs well if and only if it is equipped to make a hypothesis on the current situation. If the given situation is unaccounted for the filter breaks down. In terms of the single filter, in general the UKF displayed better performance than the EKF, especially in the case of scenario 4 where the nonlinearities of the fault caused the single EKF to breakdown. For these reasons the UKF has been chosen to develop the fault tolerant flight controller.

4.5.6 Comparison to Linear MPC

As a result of the findings given in subsection 4.5.5 only the UKF with feedback is implemented to compare nonlinear MPC with linear MPC. The results given in the next two sections are for scenarios 2 and 4 respectively.

4.5.6.1 Scenario 2

The velocity innovation plots in figure 4.85 show that the innovations remain well within the uncertainty bounds and are approximately zero. However the uncertainty is double that produced by the nonlinear controller (figure 4.22).

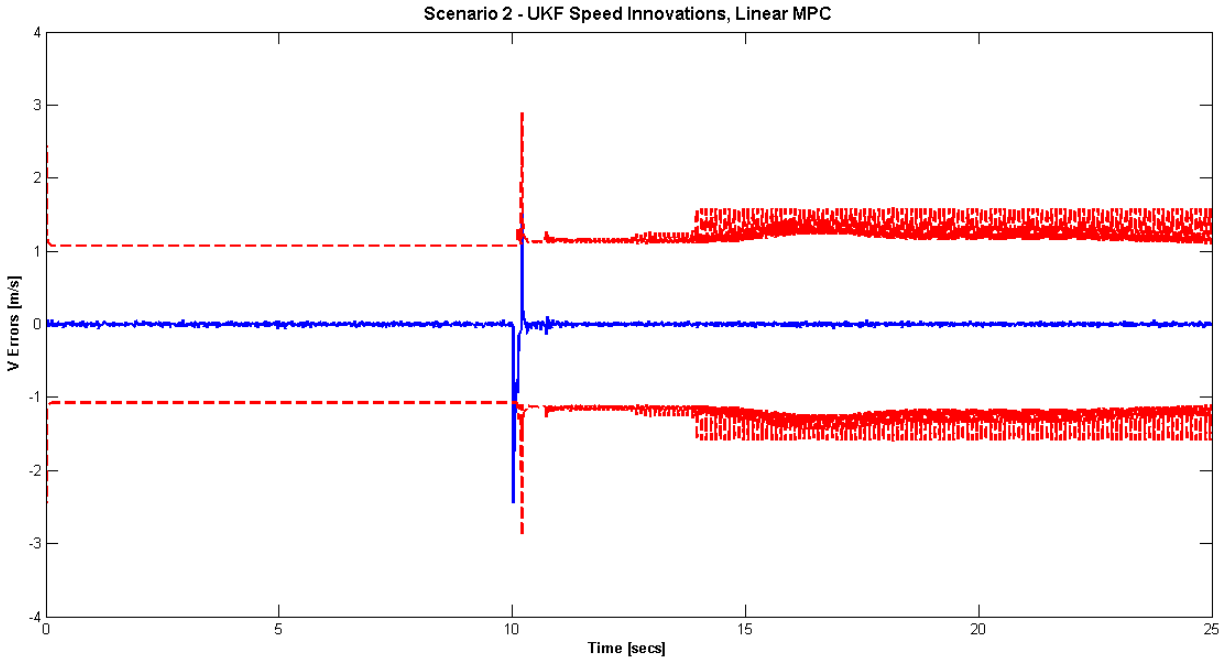

Figure 4.85: Scenario 2 - Linear MPC UKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

The wheel radii plots give in figure 4.86 show that the estimations produced by a nonlinear controller are the same as those produced by the linear controller. Hence the filter performs well even with linear MPC.

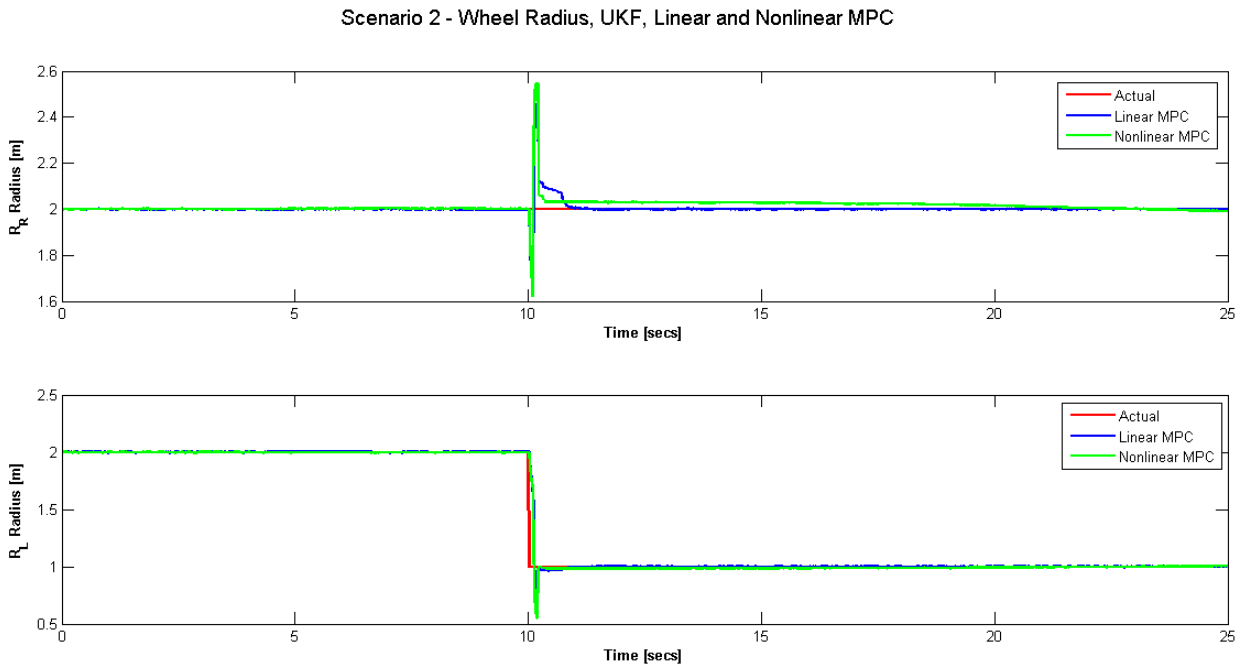

Figure 4.86: Scenario 2 - Comparison of Linear and Nonlinear MPC UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

The angular rate plots for the linear MPC controller (figure 4.87) show that five seconds after the fault occurred the linear controller pushes the wheels to operate at their constraints and is unable to tolerate the faulty condition. This is further illustrated in the trajectory plot given in figure 4.88 which clearly shows that the nonlinear MPC controller does an excellent job of keeping the robot on the path despite a faulty wheel whereas the solution produced by the linear controller has diverged.

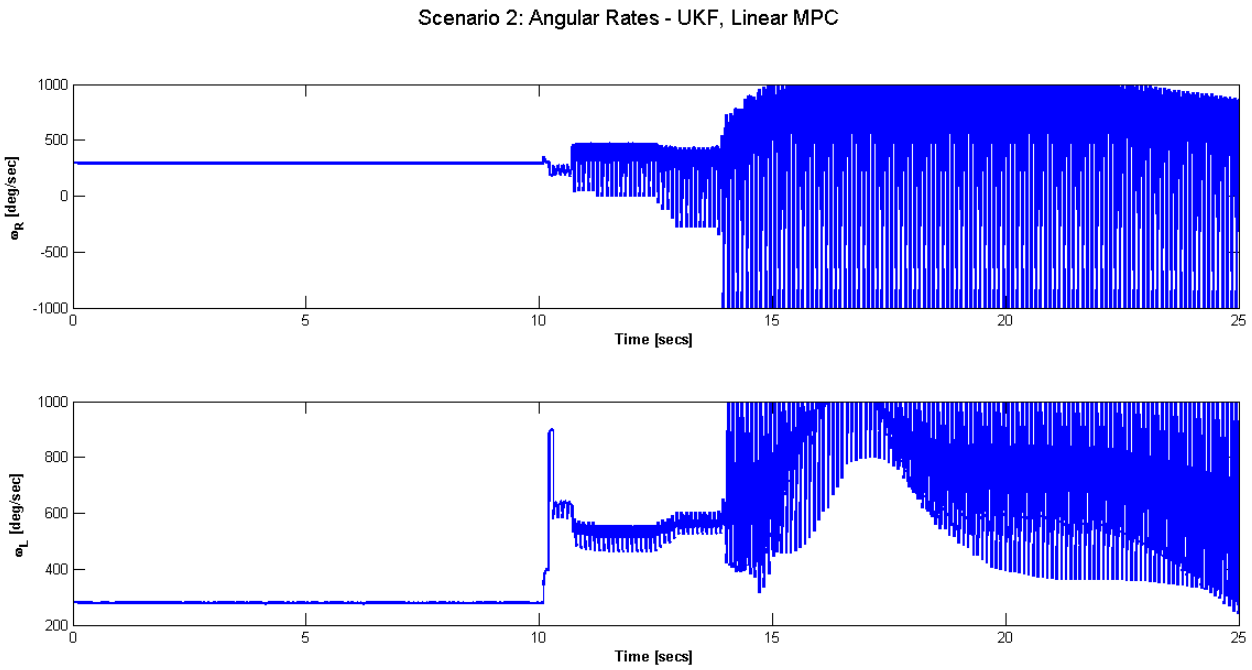

Figure 4.87: Scenario 2 - Linear MPC UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

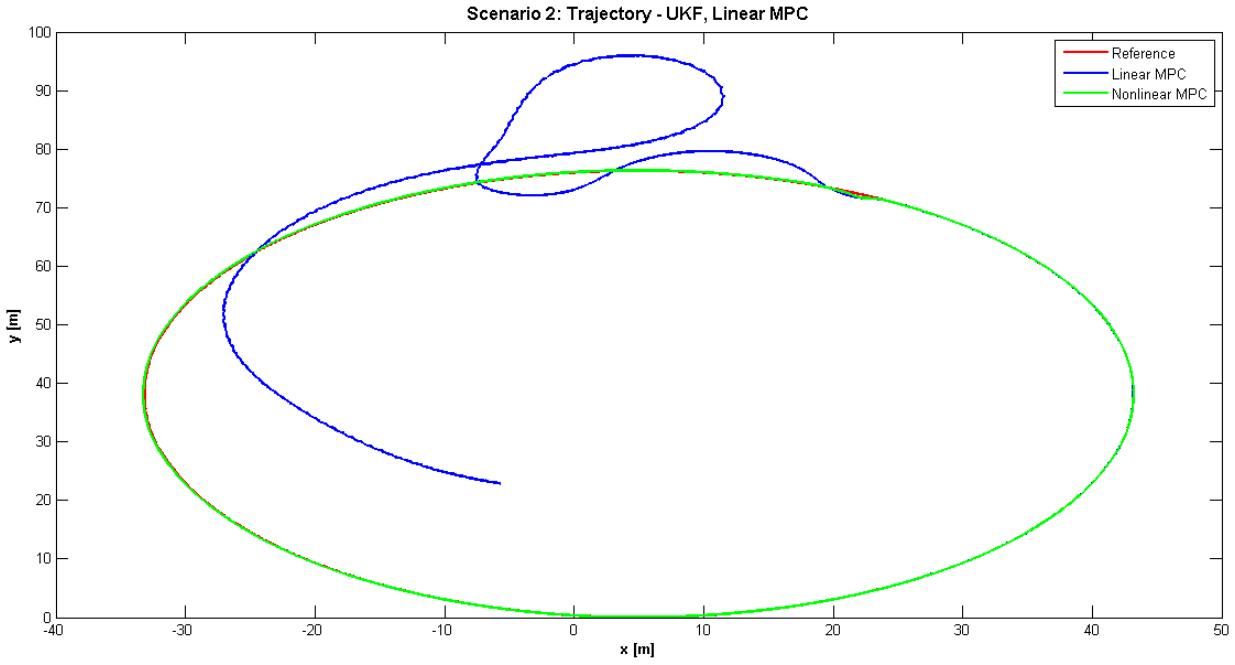

Figure 4.88: Scenario 2 - Linear and Nonlinear MPC UKF Trajectory

4.5.6.2 Scenario 4

The speed innovations plot (figure 4.89) shows similar trends to those above in that the innovations are quite small however the uncertainties with the linear filter are higher than those produced as a result of nonlinear MPC (figure 4.58). The estimates of the wheel radii however are very good and can be seen to be the same for both linear and nonlinear controllers (figure 4.92).

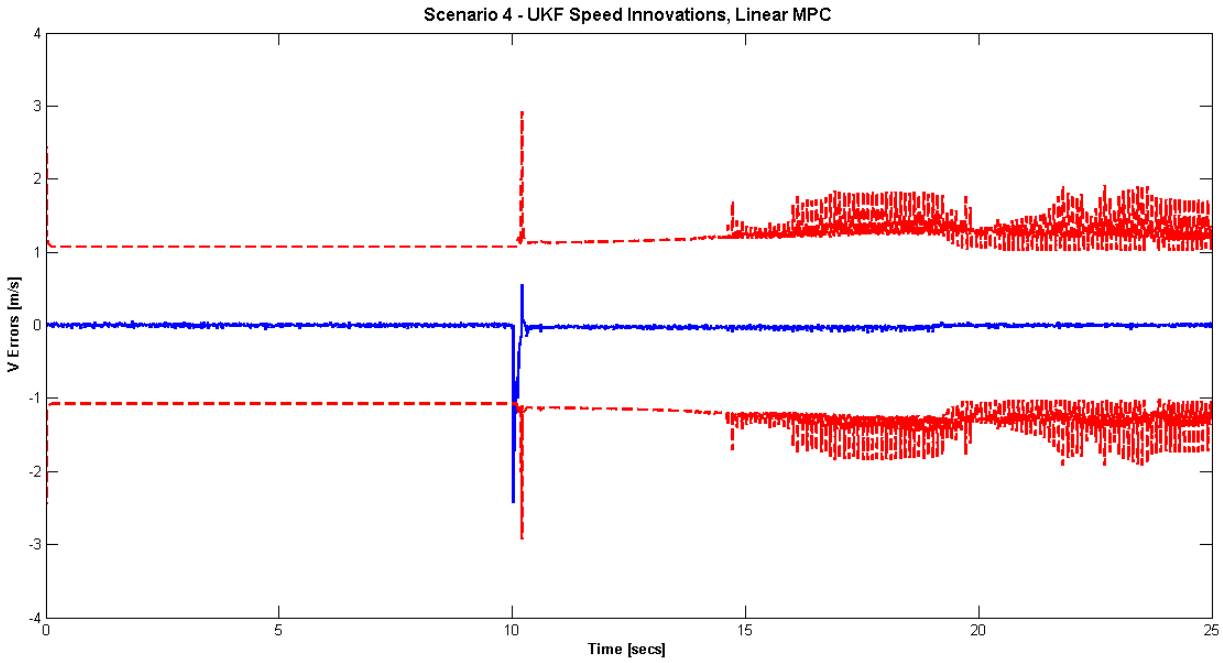

Figure 4.89: Scenario 4 - Linear MPC UKF Speed Innovations, 2σ Uncertainty Bounds (dashed lines), Speed Innovations (solid line)

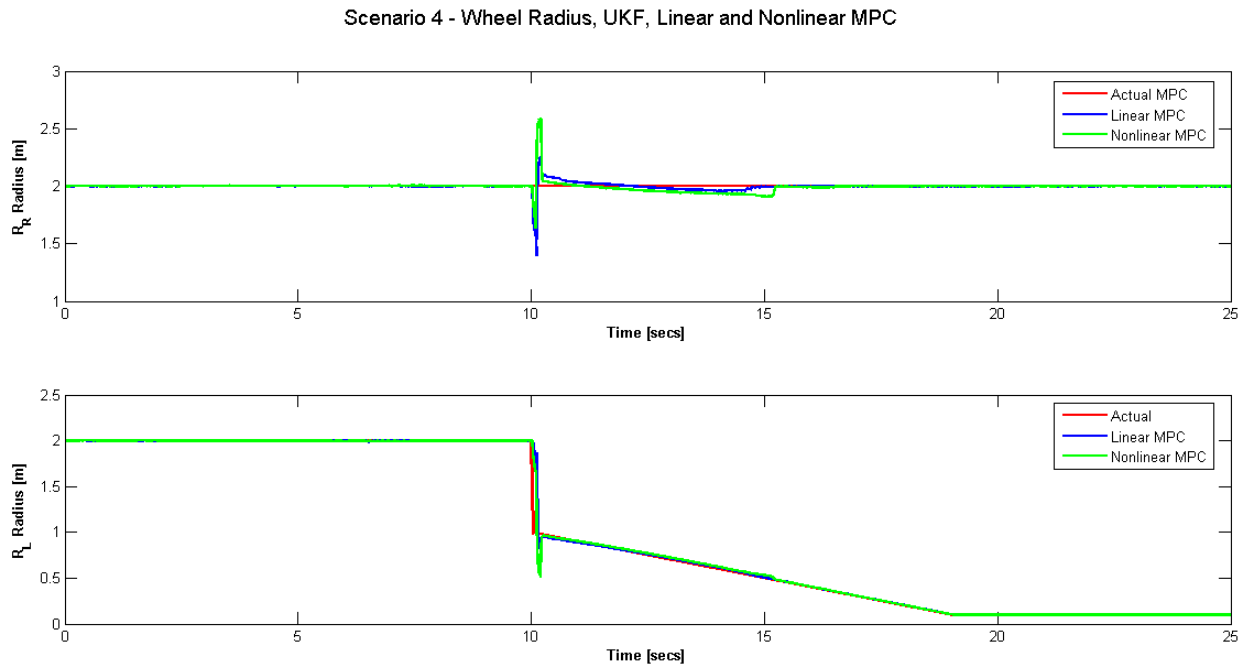

Figure 4.90: Scenario 4 - Comparison of Linear and Nonlinear MPC UKF Radius Estimates, Right wheel (top), Left wheel (Bottom)

As expected the linear controller was unable to maintain the robot on the path as is evident by the trajectory plot of figure 4.92. Neither of the controllers were able to drive the robot on the path as the wheel radius had almost reached 0m making it infeasible for the robot to continue. The angular rates given in figure 4.91 show that the linear controller constantly demands operation at the constraints oscillating between the upper and lower limits continuously. The nonlinear MPC controller results for wheel angular rates (figure 4.68) shows that the right wheel oscillates between the upper and lower bounds, however the left wheel is required to constantly work at the upper bound.

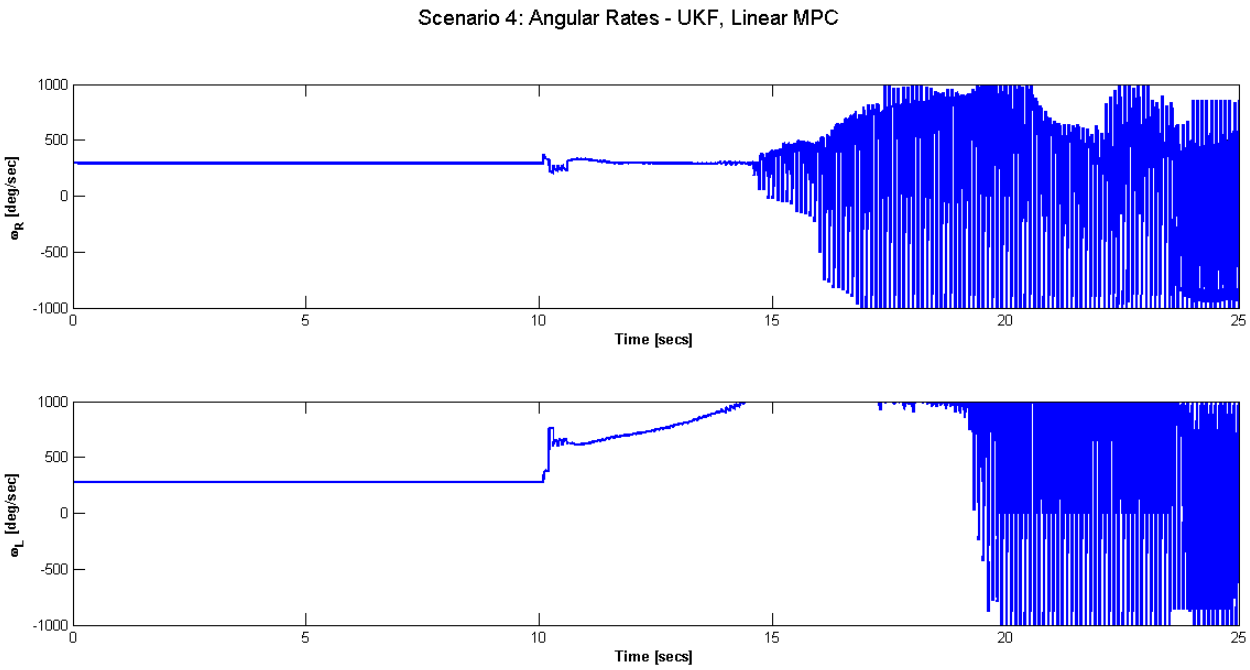

Figure 4.91: Scenario 4 - Linear MPC UKF Angular Rates, Right wheel (top), Left wheel (Bottom)

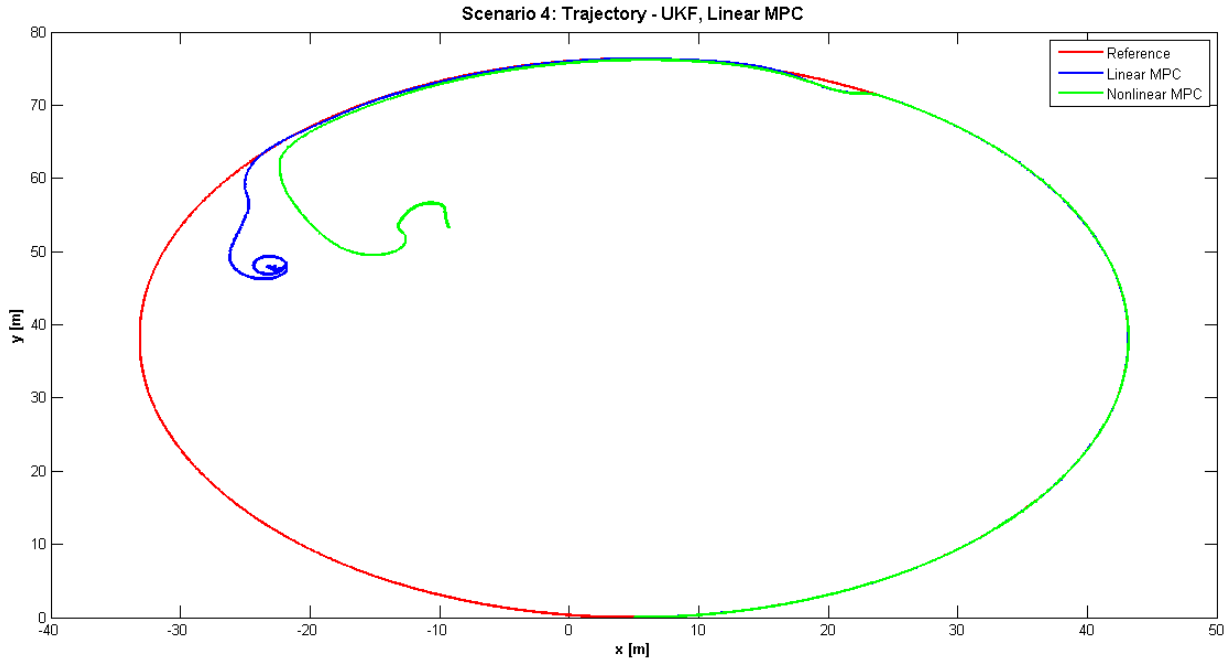

Figure 4.92: Scenario 4 - Comparison Linear and Nonlinear MPC UKF Trajectory

4.6 Summary Of Findings

The analysis from this work has proven the feasibility of my NMPC controller design with filter estimates for controller reconfiguration as a viable solution to fault tolerant control.

Comparisons were also made between the performance of nonlinear MPC and linear MPC. The results clearly show that for the purposes of reconfigurable fault tolerant control the nonlinear MPC controller has better performance.

The next chapter will implement the NMPC pseudospectral controller with a UKF based FDI subsystem to an aircraft, for fault tolerant flight control.

Chapter 5

Fault Tolerant Flight Control

5.1 Introduction

Sections 5.4 and 5.5 of this chapter have appeared as [73] and [72] respectively.

5.1.1 Motivation

In the development of any type of controller it is necessary to develop a model of the plant that is to be controlled. Plant model development is very important particularly in the design of a model predictive control (MPC) controller as a variant of the plant model also forms an integral part of the controller design. As discussed in chapter 3 a linearised model of the plant is used in the design of a linear MPC controller. Nonlinear MPC however incorporates the full nonlinear model of the plant. Hence, in this chapter, I discuss the development of the aircraft model used for both the plant and controller design.

The results of chapters 3 and 4 show that my active fault tolerant control (FTC) design using pseudospectral nonlinear model predictive control (NMPC) integrated with an unscented Kalman filter (UKF) filter is a viable solution. To address the research questions stated in chapter 2 I now apply my FTC design to aircraft control to investigate the feasibility of my design solution as a fault tolerant flight controller.

5.1.2 Outline

This chapter is dedicated to applying the knowledge gained from the application of my FTC design to the 2D robot model, to an aircraft system. To begin, an overview is given in section

5.2 including the details of the system to be modelled. It puts into context a general aircraft flight control system and gives details of what will be developed for this research and where my system will fit into the overall aircraft flight control system.

This is followed by a section on aircraft dynamics (section 5.3), which outlines the equations used to develop the plant model and ultimately the prediction model of the NMPC controller.

The development of the FTC system is divided into two parts, longitudinal motion (section 5.4) followed by the development of the full 6DoF model (section 5.5) which involves combining both the longitudinal and lateral motions of the aircraft. The build up to the combined motion model is required because aircraft motion is highly complex. These two sections look at controller development only and assume fault detection and identification (FDI) information is provided.

The design process of the FDI subsystem is depicted in section 5.6 for the full 6DoF aircraft model. The research results within this section show that a more thorough examination of this area is required as many problems were encountered, most beyond the scope of this current research. To remain within the scope of this research and to demonstrate the proof of concept of the feasibility of my FTC system design to flight control, section 5.7 looks at a very specific application, namely engine failure during longitudinal motion. Engine failure results in a loss of power to the aircraft reducing its ability to maintain flight. Thus, section 5.7 develops the full fault tolerant controller plus FDI system for flight control.

Finally the findings are summarised in section 5.8 and a conclusion given, based on the results.

5.2 System Overview

All autonomous unmanned aerial vehicles (UAVs) are equipped with an onboard guidance, navigation and control (GNC) system. The system flies the aircraft on the desired trajectory. Thus the design of a fault tolerant controller necessarily starts with an understanding of a typical UAV GNC system model (given in figure 5.1). A GNC system usually comprises of an outer loop subsystem which is provided with a set of way points (points used to define an aircraft trajectory) and navigation information. The navigation subsystem provides information on the current status of the aircraft and uses filtering techniques to estimate aircraft position, orientation, velocities and angular rates based on GPS data and sensor data provided by the plant.

Sensor information consists of data, including linear accelerations and angular rates, provided by the inertial measurement unit on board the aircraft. The outer loops compare the current position of the aircraft with the required position based on the waypoints, and calculate a set of demands (typically velocity and angular rate demands) that are then passed to the inner loops. Navigation information as well as the demands from the outer loops is supplied to the inner loops, and based on this, the flight controller, which sits within the inner-loops subsystem, calculates the necessary control inputs required to achieve the demands. The control inputs are then applied to the aircraft (plant) and sensor information is fed to the navigation subsystem and the whole process begins again.

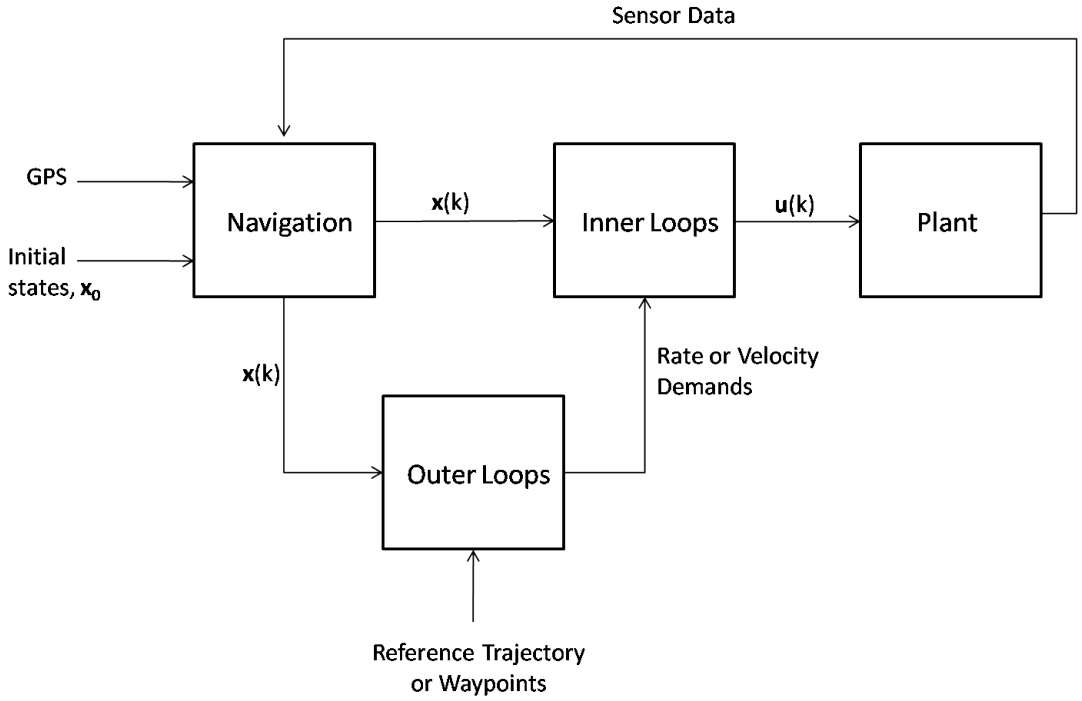

Figure 5.1: System Block Diagram of a Typical Guidance and Navigation Loop

My objective is the development of an active fault tolerant flight control system. Hence the typical guidance and navigation system given in figure 5.1 becomes the system given in figure 5.2. The fault tolerant control system, the area enclosed by the dashed rectangle and comprising of the innerloops and the FDI subsystems, is the focus of my research. The NMPC controller developed in chapter 3 sits within the inner-loops subsytem; and the filter used for fault detection lies inside the FDI subsystem.

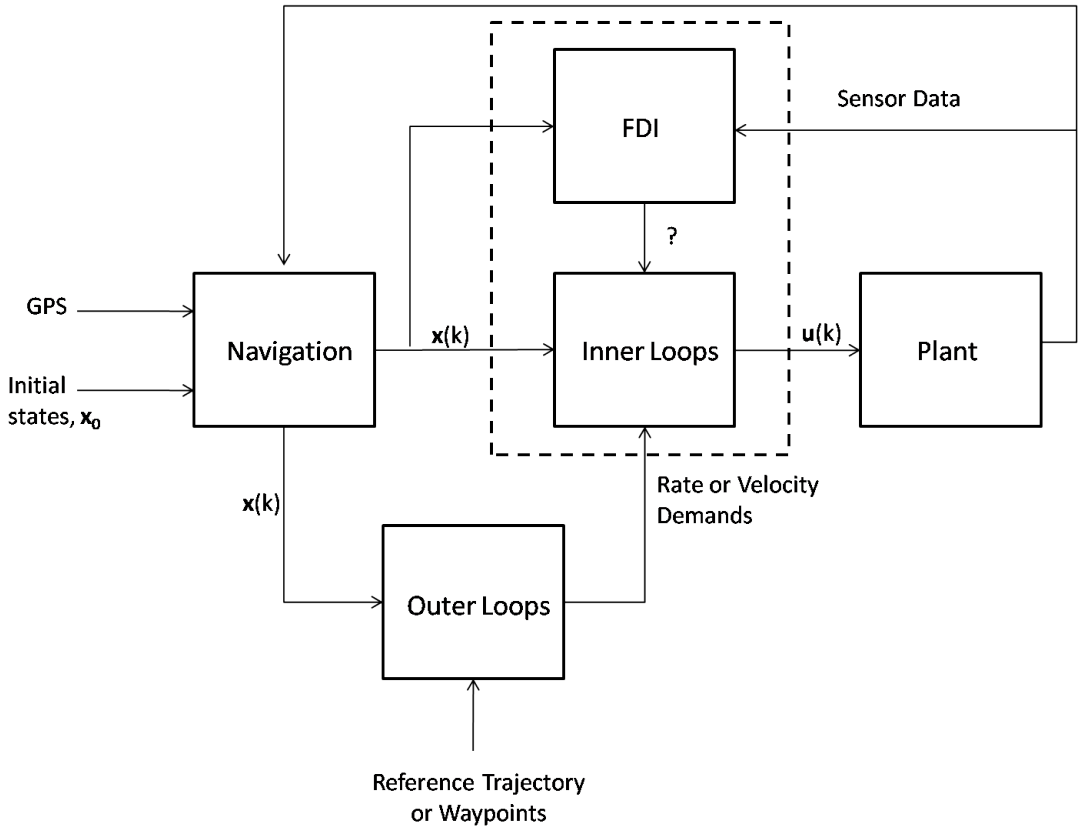

Figure 5.2: System Block Diagram of a Guidance and Navigation Loop with an Active Fault Tolerant Flight Control System

As is evident from the diagrams in figures 5.1 and 5.2, it is necessary to derive a plant model to develop the controller. Here, the navigation and guidance subsystems are not developed but assumed. The plant model doubles as the navigation model and the guidance subsystem for the 6DoF work in section 5.5 is provided by Williams [127]. The next section details the aircraft model.

5.3 Aircraft Model

The aircraft model to be used as the plant model and ultimately the prediction model for the NMPC controller is developed in this section. For any control system design the behaviour of the system to be controlled, in this case an aircraft, is simulated by the plant model. Thus, for flight controller development the plant must simulate an aircraft, governed by the equations of motion for flight.

The equations of motion for an aircraft are well defined and well documented [115] [114]. The

orientation of an aircraft with respect to the airflow directly affects the forces and moments produced on the body of the aircraft which act at the centre of gravity. In a uniform airflow the aerodynamic forces and moments are unchanged after a rotation around the freestream velocity vector, and only two orientation angles with respect to the relative wind are needed to calculate them. These angles, known as aerodynamic angles, the angle of attack α and the angle of sideslip β, are defined within the body-fixed coordinate frame denoted by the subscript **b**, (see figure 5.3).

Aircraft motion is defined by position coordinates, linear velocities, orientation angles and angular velocities (or angular rates). The state vector of the aircraft plant model is therefore given by:

$$\mathbf{x} = [x_N \;\; x_E \;\; x_D \;\; V_N \;\; V_E \;\; V_D \;\; \phi \;\; \theta \;\; \psi \;\; p \;\; q \;\; r]^\mathsf{T}, \tag{5.1}$$

where x_N, x_E, x_D are north, east and down position coordinates respectively, given in an earth fixed tangent frame, called the navigation frame (or North, East, Down (NED) frame), denoted by the subscript **n**, (see figure 5.3). The NED is an Earth fixed frame, with the origin located at a point on the Earth. In practice this origin is defined at the point where the aircraft is initialised for flight. The vectors V_N, V_E, V_D are the velocities in the north, east, and down directions respectively, ϕ, θ, ψ are the aircraft orientation angles roll, pitch and yaw respectively and, p, q, r are the roll, pitch and yaw angular rates respectively.

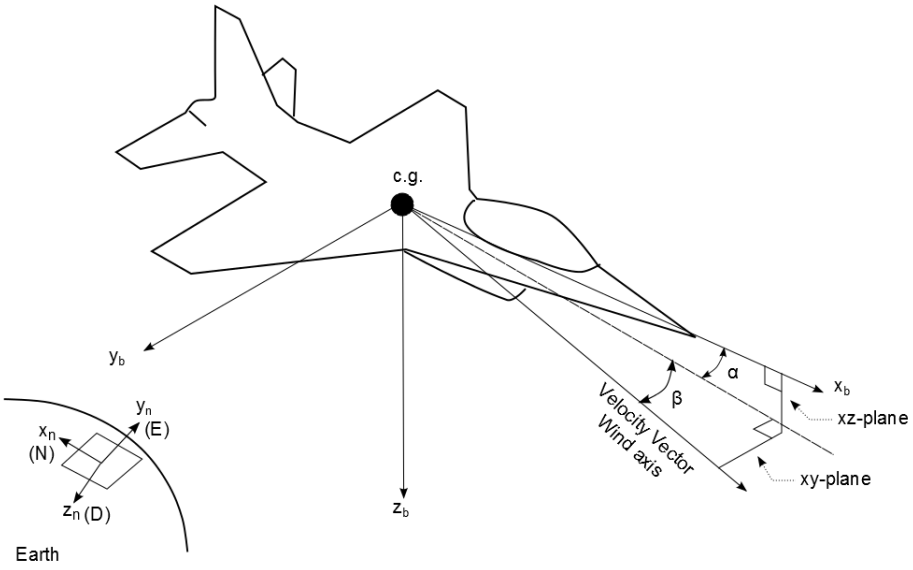

Figure 5.3: Aircraft Coordinate Frames

The forces and moments used to derive the equations of motion act at the centre of gravity (c.g.) of the aircraft, which is also the location of the body fixed coordinate frame. All calculations are performed in the body axis, with the position and velocities in the body axis then converted to the NED frame. A conversion via multiplication with a direction cosine matrix (DCM) [115], transforms the vectors in one axis system to another. The direction cosine matrix, C_b^n [115], is used to transform the body axis to the NED frame and is given by a 3-2-1 right-handed rotation sequence, namely [115]:

- Right-handed rotation about the z-axis (positive ψ),
- Right-handed rotation about the new y-axis (positive θ),
- Right-handed rotation about the new x-axis (positive ϕ).

$$C_b^n = \begin{bmatrix} \cos\theta\cos\psi & -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}. \tag{5.2}$$

Note that, to go from the NED frame to the body frame the required DCM is the transpose of C_b^n:

$$C_n^b = (C_b^n)^\intercal . \tag{5.3}$$

Up until this point the state vector of the plant has been described. The plant model is initialised with information in the form of equation (5.1) and the plant model must also provide information in the same form. The first three elements of the state vector, x_N, x_E and x_D are propagated with time as follows:

$$\dot{x}_N = V_N, \tag{5.4}$$

$$\dot{x}_E = V_E, \tag{5.5}$$

$$\dot{x}_D = V_D. \tag{5.6}$$

The next three elements V_N, V_E and V_D are calculated via:

$$\dot{V}_N = a_N, \tag{5.7}$$

$$\dot{V}_E = a_E, \tag{5.8}$$

$$\dot{V}_D = a_D + g. \tag{5.9}$$

Here a_N, a_E and a_D are the accelerations in the navigation frame, and g is acceleration due to gravity equal to $9.81 m/s^2$. All calculations are executed in the body axis hence the accelerations are firstly found in the body axis then converted to the navigation frame. The body accelerations a_X, a_Y and a_Z are calculated via [52]:

$$a_X = \frac{\bar{q}SC_X + T}{m}, \qquad a_Y = \frac{\bar{q}SC_Y}{m}, \qquad a_Z = \frac{\bar{q}SC_Z}{m}, \tag{5.10}$$

where $\bar{q}$ is dynamic pressure and is given by:

$$\bar{q} = \frac{1}{2}\,\rho\, {V_T}^2, \tag{5.11}$$

ρ is air density and has been taken at sea level which is $1.225\, kg/m^3$, S is the aircraft wing area and m is the mass of the aircraft. In the acceleration equations T is the total thrust and it is assumed that the thrust acts along the X body-axis. The terms C_X, C_Y and C_Z are the non-dimensional forces in the X, Y and Z directions, respectively, in the body coordinate frame. The calculation of these forces vary from aircraft to aircraft but are always a function of the angle of attack, α and the sideslip angle β. The aerodynamic angles are calculated as follows:

To begin, the velocities from the state vector are converted from the navigation frame to the body axis and are denoted by u, v and w in the x_b, y_b and z_b directions respectively.

$$[u,\ \ v,\ \ w]^\intercal = C_n^b\ [v_{N_{\text{rel}}},\ \ v_{E_{\text{rel}}},\ \ v_{D_{\text{rel}}}]^\intercal\,. \tag{5.12}$$

Where the subscript 'rel' indicates velocity relative to the wind. The angles α and β are given by [52]:

$$\alpha \ = \ \arctan\left(\frac{w}{u}\right), \tag{5.13}$$

$$\beta \ = \ \arctan\left(\frac{v}{V_T}\right), \tag{5.14}$$

where the true airspeed, V_T, is given by:

$$V_T = \sqrt{(v_{N_{\text{rel}}})^2 + (v_{E_{\text{rel}}})^2 + (v_{D_{\text{rel}}})^2}. \tag{5.15}$$

The effects of wind and turbulence are considered in the plant model hence V_{rel} is given by [115]:

$$\begin{aligned} V_{\text{rel}} \ &= \ v_{\text{nav}} - v_{\text{wind}}, \\ &= \ [v_N\ \ v_E\ \ v_D]^\intercal - [v_{N_{\text{wind}}}\ \ v_{E_{\text{wind}}}\ \ v_{D_{\text{wind}}}]^\intercal\,, \\ &= \ [v_{N_{\text{rel}}}\ \ v_{E_{\text{rel}}}\ \ v_{D_{\text{rel}}}]^\intercal\,, \end{aligned} \tag{5.16}$$

and ω_{rel} is given by:

$$\begin{aligned}\omega_{\text{rel}} &= \omega + \omega_{\text{wind}}, \\ &= [p \;\; q \;\; r]^{\intercal} + [p_{\text{wind}} \;\; q_{\text{wind}} \;\; r_{\text{wind}}]^{\intercal}, \\ &= [p_{\text{rel}} \;\; q_{\text{rel}} \;\; r_{\text{rel}}]^{\intercal}, \end{aligned} \tag{5.17}$$

where V_{NED} is the aircraft velocity vector, V_{wind} is the wind velocity vector with both being measured in the NED coordinate frame. ω is the vector of angular rates of the aircraft and ω_{wind} is the angular rate (or turbulence) of the wind.
Finally the acceleration in the navigation frame is given by:

$$[a_N, \;\; a_E, \;\; a_D]^{\intercal} = [\dot{v}_N, \;\; \dot{v}_E, \;\; \dot{v}_D]^{\intercal} \tag{5.18}$$

$$= C_b^n \; [a_X, \;\; a_Y, \;\; a_Z]^{\intercal} + [0, \;\; 0, \;\; g]^{\intercal}. \tag{5.19}$$

The Euler angle (or aircraft orientation angles), ϕ (roll), θ (pitch) and ψ (yaw) are the next three elements in the state vector (equation (5.1)). These angles are given in the body axis and are not converted to the NED frame. They are propagated in time via [52]:

$$\dot{\phi} = p + \tan\theta \left(q \sin\phi + r \cos\phi\right), \tag{5.20}$$

$$\dot{\theta} = q \cos\phi - r \sin\phi, \tag{5.21}$$

$$\dot{\psi} = \frac{q \sin\phi + r \cos\phi}{\cos\phi}. \tag{5.22}$$

Finally the last three components of the state vector, equation (5.1), are the body angular rates p, q and r calculated using the following equations [52]:

$$\dot{p} = \left(c_1 \, r + c_2 \, p + c_4 \, h_{eng}\right) \, q + \bar{q} \, S \, b \left(c_3 \, C_l + c_4 \, C_n\right), \tag{5.23}$$

$$\dot{q} = \left(c_5 \, p - c_7 \, h_{eng}\right) r - c_6 \left(p^2 - r^2\right) + \bar{q} \, S \, \bar{c} \, c_7 \, C_m, \tag{5.24}$$

$$\dot{r} = \left(c_8 \, p - c_2 \, r + c_9 \, h_{eng}\right) q + \bar{q} \, S \, b \left(c_4 \, C_l + c_9 \, C_n\right), \tag{5.25}$$

where:

$$c_1 = \frac{\left(I_Y - I_Z\right) I_Z - {I_{XZ}}^2}{I_X I_Z - {I_{XZ}}^2}, \qquad c_2 = \frac{\left(I_X - I_Y + I_Z\right) I_{XZ} - {I_{XZ}}^2}{I_X I_Z - {I_{XZ}}^2}, \qquad c_3 = \frac{I_Z}{I_X I_Z - {I_{XZ}}^2},$$

$$c_4 = \frac{I_{XZ}}{I_X I_Z - {I_{XZ}}^2}, \qquad c_5 = \frac{I_Z - I_X}{I_Y}, \qquad c_6 = \frac{I_{XZ}}{I_Y},$$

$$c_7 = \frac{1}{I_Y}, \qquad c_8 = \frac{\left(I_x - I_Y\right) - {I_{XZ}}^2}{I_X I_Z - {I_{XZ}}^2}, \qquad c_9 = \frac{I_X}{I_X I_Z - {I_{XZ}}^2}.$$

The I terms are known as moments of inertia and C_l, C_m and C_n are known as moment coefficients in roll, pitch and yaw respectively and vary from aircraft to aircraft. The term h_{eng} is the distance of the engine from the aircraft c.g. and for this work it is assumed to be zero. The parameter $\bar{c}$ is known as the mean aerodynamic chord of the wing. The values for p, q and r are provided as sensor measurements from the inertial measurement unit.

The general structure of the equations for the the non-dimensional force and moment coefficients C_X, C_Y, C_Z, C_l, C_m, C_n can be found in Stevens and Lewis [115]. The next section will discuss in further detail the aircraft specific values used in this research.

5.3.1 Aircraft Specific Data

The equations of motion given thus far are the general equations of motion for a 6DoF aircraft model. To develop the fault tolerant controller a specific aircraft model was required. A collection of nonlinear aircraft simulation models complete with full mathematical equations, for a number of aircraft, have been provided by NASA [52] in the open literature for research purposes. The authors, Garza and Morelli [52], recognise that nonlinear aircraft simulations are useful for dynamic analysis, control law design and validation, guidance and trajectory studies, air combat investigations, pilot training, and many other tasks [52] and hence have provided this valuable tool. All models given in [52] are based on manned aircraft, however the equations supplied are non-dimensional, hence they can easily be modified represent a compact aircraft such as a UAV.

The generic aircraft model developed here for control law design and validation is based on the McDonnell Douglas F-4 aircraft [52]. The aircraft specific data used for the modelling is summarised next.

5.3.1.1 Dimensions and Weights

The aircraft model used for simulations and analysis is a fictional model with the aerodynamic characteristics of the F-4 Phantom [52]. The properties given in tables 5.1 and 5.2 were used for the simulated aircraft model.

Table 5.1: Simulated Aircraft Dimensional Properties

Wing Area S	20 m
Mean Aerodynamic Chord $\bar{c}$	3 m
C.G location $x_{c.g}$	0 m
C.G reference location $x_{c.g.ref}$	0 m
Stall speed V_{stall}	23 m/s

Table 5.2: Mass Properties of model used for simulation, in SI Units

Parameter	Weight (kg)	I_X (kg.m^2)	I_Y (kg.m^2)	I_Z (kg.m^2)	I_{XZ} (kg.m^2)
Value	1,177	2,257	11,044	12,636	106

The non-dimensional force and moment equations pertaining to the aircraft model can now be detailed.

5.3.1.2 Force and Moment Coefficients

As previously mentioned the fictional aircraft has the aerodynamic characteristics (force and moment properties) of the F-4 Phantom. At $\alpha \leq 15^0$ the non-dimensional force and moment coefficients are given by [52]:

$$
\begin{aligned}
C_X = & -0.0434 + 2.93 \times 10^{-3}\alpha + 2.53 \times 10^{-5}\beta^2 - 1.07 \times 10^{-6}\alpha\beta^2 + 9.5 \times 10^{-4}\delta_e \\
& - 8.5 \times 10^{-7}\delta_e\beta^2 + \left(\frac{180q\bar{c}}{\pi 2V_t}\right)\left(8.73 \times 10^{-3} + 0.001\alpha - 1.75 \times 10^{-4}\alpha^2\right),
\end{aligned} \tag{5.26}
$$

$$
\begin{aligned}
C_Y = & -0.012\beta + 1.55 \times 10^{-3}\delta_r - 8 \times 10^{-6}\delta_r\alpha \\
& + \left(\frac{180b}{\pi 2V_t}\right)\left(2.25 \times 10^{-3}p + 0.0117r - 3.67 \times 10^{-4}r\alpha + 1.75 \times 10^{-4}r\delta_e\right),
\end{aligned} \tag{5.27}
$$

$$
\begin{aligned}
C_Z = & -0.131 - 0.0538\alpha - 4.76 \times 10^{-3}\delta_e - 3.3 \times 10^{-5}\delta_e\alpha - 7.5 \times 10^{-5}{\delta_a}^2 \\
& + \left(\frac{180q\bar{c}}{\pi 2V_t}\right)\left(-0.111 + 5.17 \times 10^{-3}\alpha - 1.1 \times 10^{-3}\alpha^2\right),
\end{aligned} \tag{5.28}
$$

$$
\begin{aligned}
C_l = & -5.98 \times 10^{-4}\beta - 2.83 \times 10^{-4}\alpha\beta + 1.51 \times 10^{-5}\alpha^2\beta \\
& - \delta_a\left(6.1 \times 10^{-4} + 2.5 \times 10^{-5}\alpha - 2.6 \times 10^{-6}\alpha^2\right) \\
& + \delta_r\left(-2.3 \times 10^{-4} + 4.5 \times 10^{-6}\alpha\right) \\
& + \left(\frac{180b}{\pi 2V_t}\right)\left(-4.2 \times 10^{-3}p - 5.24 \times 10^{-4}p\alpha + 4.36 \times 10^{-5}p\alpha^2\right. \\
& \left. + 4.36 \times 10^{-4}r + 1.05 \times 10^{-4}r\alpha + 5.24 \times 10^{-5}r\delta_e\right),
\end{aligned} \tag{5.29}
$$

$$
\begin{aligned}
C_m = & -6.61 \times 10^{-3} - 2.67 \times 10^{-3}\alpha - 6.48 \times 10^{-5}\beta^2 \\
& - 2.65 \times 10^{-6}\alpha\beta^2 - 6.54 \times 10^{-3}\delta_e - 8.49 \times 10^{-5}\delta_e\alpha \\
& + 3.74 \times 10^{-6}\delta_e\beta^2 - 3.5 \times 10^{-5}{\delta_a}^2 \\
& + \left(\frac{180q\bar{c}}{\pi 2V_t}\right)\left(-0.0473 - 1.57 \times 10^{-3}\alpha\right) + \left(x_{c.g.ref} - x_{c.g}\right)C_Z,
\end{aligned} \tag{5.30}
$$

$$
\begin{aligned}
C_n = & \, 2.28 \times 10^{-3}\beta + 1.79 \times 10^{-6}\beta^3 + 1.4 \times 10^{-5}\delta_a \\
& + 7.0 \times 10^{-6}\delta_a\alpha - 9.0 \times 10^{-4}\delta_r + 4.0 \times 10^{-6}\delta_r\alpha \\
& + \left(\frac{180b}{\pi 2V_t}\right)\left(-6.63 \times 10^{-5}p - 1.92 \times 10^{-5}p\alpha + 5.06 \times 10^{-6}p\alpha^2\right. \\
& \qquad \left. -6.06 \times 10^{-3}r - 8.73 \times 10^{-5}r\delta_e + 8.7 \times 10^{-6}r\delta_e\alpha\right) \\
& - \left(\frac{\bar{c}}{b}\right)\left(x_{c.g.ref} - x_{c.g}\right)C_Z.
\end{aligned} \tag{5.31}
$$

The δ_a, δ_e and δ_r terms are the control surface deflection angles, with details given in 5.3.2.

Note: In the plant model when calculating the forces and moments the velocity and angular rates are as given by equations (5.16) and (5.17) respectively, ie. the effects of wind and turbulence are taken into account. However the prediction model used for the NMPC has no knowledge of these disturbances and the wind and turbulence component is zero.

5.3.1.3 Thrust Model

The thrust force required to keep an aircraft in motion is generated by the engines. The thrust term, T, appears in the velocity equations (5.10). The following thrust model taken from Bryson [25] is used here:

$$h_T = \frac{H}{3048}, \tag{5.32}$$

$$\begin{aligned} T_{max} = ((&30.21 - 0.668\,h_T - 6.877\,{h_T}^2 + 1.951\,{h_T}^3 - 0.1512\,{h_T}^4) \\ &+ \left(\frac{V_T}{v_s}\right)(-33.8 + 3.347\,h_T + 18.13\,{h_T}^2 - 5.865\,{h_T}^3 + 0.4757\,{h_T}^4) \\ &+ \left(\frac{V_T}{v_s}\right)^2(100.8 - 77.56\,h_T + 5.441\,{h_T}^2 + 2.864\,{h_T}^3 - 0.3355\,{h_T}^4) \\ &+ \left(\frac{V_T}{v_s}\right)^3(-78.99 + 101.4\,h_T - 30.28\,{h_T}^2 + 3.236\,{h_T}^3 - 0.1089\,{h_T}^4) \\ &+ \left(\frac{V_T}{v_s}\right)^4(18.74 - 31.6\,h_T + 12.04\,{h_T}^2 - 1.785\,{h_T}^3 + 0.09417\,{h_T}^4))\frac{4448.22}{20}, \end{aligned} \tag{5.33}$$

$$T = T_{max}\,\delta_{th}, \tag{5.34}$$

where v_s is the speed of sound, $340.3\,m/s$, H the height or the $-x_D$ position of the aircraft, and δ_{th}, the amount of throttle applied, forms a part of the control vector (explained in subsection 5.3.2).

The derivatives of the states are integrated using fourth order Runge-Kutta integration to obtain the position, orientation, speeds and angular rates of the aircraft.

The next section describes the aircraft controls and defines the terms δ_{th}, δ_e, δ_a and δ_r.

5.3.2 Aircraft Controls

The purpose of any control system is to manipulate the available controls to achieve a desired outcome. For an aircraft the available controls are effected through the manipulation of the con-

trol surfaces. Control surfaces are manipulated by deflecting them to achieve a desired response and it is the job of the control system to calculate the amount of deflection required.

The three most common control surfaces are the elevators, ailerons and the rudder. Figure 5.4 shows the location of these control surfaces on an aircraft. The elevators are used to control pitch (longitudinal motion), resulting in the nose of the aircraft going up or down. The ailerons control roll movement causing one side of the aircraft wing to go down while the other goes up. Finally the rudder controls the yaw angle of the aircraft, moving the nose of the aircraft right or left. The rudder and the aileron together control the lateral motion of the aircraft

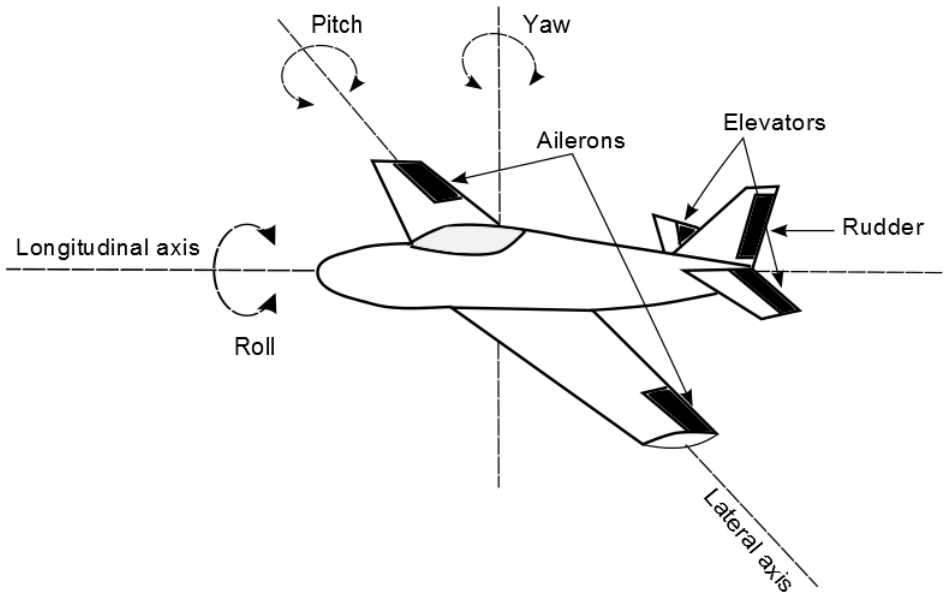

Figure 5.4: Aircraft Control Surfaces

Another very important control input is the throttle, not given in the diagram as it is not located on the outside body of the aircraft. The throttle controls the amount of thrust required with the amount of throttle applied varying from 0% to 100%.

The equations given for the force and moment coefficients in section 5.3.1.2 contain the terms δ_e, δ_a and δ_r. These terms represent the deflection angles for the elevator, aileron and rudder respectively. The controls for the aircraft model are throttle (δ_{th}), δ_e, δ_a and δ_r, and the control vector is given by:

$$\mathbf{u} = [\delta_{th},\ \delta_a,\ \delta_e,\ \delta_r]^\intercal . \tag{5.35}$$

The sign convention used for the control surfaces is given in 5.3:

Table 5.3: Control Surface sign conventions.

Elevator	trailing edge down	positive	negative pitching moment
Aileron	right-wing trailing edge down	positive	negative rolling moment
Rudder	trailing edge left	positive	negative yawing moment

The equations presented above are used to model the plant in MATLAB/Simulink. Details of the use of these equations in the modelling of the aircraft FTC system are presented next.

5.4 Controller Design for Longitudinal Motion

Parts of this section have appeared as [73].

The design process of the FTC controller is split into two parts. Initially a controller for only the longitudinal motion of the aircraft is designed and then the model extended to the full 6DoF model, presented in section 5.5.

5.4.1 Model Description

The system given in figure 5.2 is modelled in MATLAB/Simulink. The aircraft equations given in section 5.3 are used, for the plant and the prediction model for the NMPC controller. Note, however that when investigating longitudinal motion the following values are set to zero: δ_a, δ_r, p, r, C_Y, C_l, C_n, β, ϕ and ψ. The plant doubles as a navigation system and guidance information is assumed to be received in the form of a reference trajectory. Trajectory information includes a desired height profile, required vertical speed and true airspeed. The effect of wind is not considered for this part of the investigation.

The equations of motion are integrated forward in the plant model using a Runge-Kutta integration method with the MATLAB subroutine ode45. The controller runs at 10Hz and the equations of motion are used as constraints to the optimal control problem. A pseudospectral discretisation method is used with 50 collocation points and a prediction window H_p of 5 secs.

The optimal control inputs are calculated via SNOPT. The aircraft is required to follow the trajectory given in figure 5.5

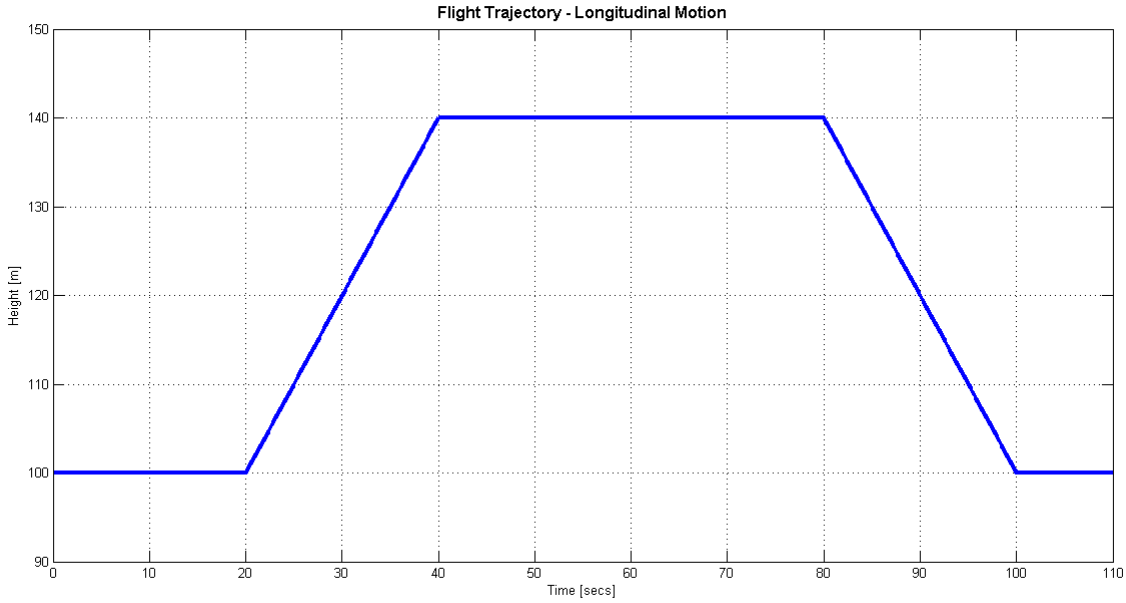

Figure 5.5: Flight Trajectory for Longitudinal Motion

Adopting the pseudospectral discretisation method where both the states and controls are discretised, the NMPC optimisation vector is:

$$\mathbf{x_{nmpc}} = [x_D,\ V_N,\ V_D,\ \theta,\ q,\ \delta_{th},\ \delta_e,\ \Delta\delta_{th},\ \Delta\delta_e]^\intercal\,, \tag{5.36}$$

where $\Delta\delta_{th}$ is the rate of change of the throttle δ_{th} and $\Delta\delta_e$ is the rate of change of the elevator deflection δ_e.

The following optimal control problem is then solved:

$$J = \min_{\mathbf{x},\mathbf{u}} \frac{H_p}{2} \sum_{j=1}^{j=N+1} \left(\left\|\mathbf{x}_D(j) - \mathbf{x}_{D_{\text{ref}}}(j)\right\|^2_{Q_x} + \left\|\mathbf{V}_t(j) - \mathbf{V}_{t_{\text{ref}}}(j)\right\|^2_{Q_V} + \left\|\Delta\delta_e\right\|^2_{Q_u} \right) w(j), \tag{5.37}$$

subject to

$$\left(\frac{t_f - t_0}{2}\right)\mathbf{D}_{j,k}\mathbf{x}\,(t) - \dot{\mathbf{x}} = 0, \tag{5.38}$$

$$\mathbf{x}(t_0) - \mathbf{x}_{\mathrm{dem}}(t_0) = 0, \tag{5.39}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{5.40}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}, \tag{5.41}$$

$$\Delta\delta_{e_{\mathrm{lb}}} \leq \Delta\delta_e \leq \Delta\delta_{e_{\mathrm{ub}}}, \tag{5.42}$$

where $\mathbf{x}_D$ and $\mathbf{x}_{D_{\mathrm{ref}}}$ are the actual and reference heights respectively, and $\mathbf{V}_t$ and $\mathbf{V}_{t_{\mathrm{ref}}}$ are the actual and reference true airspeeds respectively. The cost function weights are square matrices with the following diagonal values for each state: $Q_x = 10$, $Q_V = 5$ and $Q_u = 0.001$ The constraints applied are given in table 5.4.

Table 5.4: Constraints for longitudinal Motion

Variable	**Upper Constraint**	**Lower Constraint**
x_D	300 m	1 m
V_N	100 m/s	30 m/s
V_D	3 m/s	−3 m/s
θ	None	None
q	None	None
δ_e	20 deg	−20 deg
δ_{th}	100%	0%
$\Delta\delta_{th}$	200 %/s	−200 %/s
$\Delta\delta_e$	200 deg/s	−200 deg/s

Note: the Control surface rates given in table 5.4 are realistic for a high performance unstable airframe or for a lower weight aircraft with a stable airframe, in either case a feasible fictional aircraft model has been produced for simulation purposes to demonstrate proof of concept.

Given the above constraints in the event of no fault, the amount of throttle required is shown in figure 5.6, for the aircraft to maintain the height profile given in in figure 5.5.

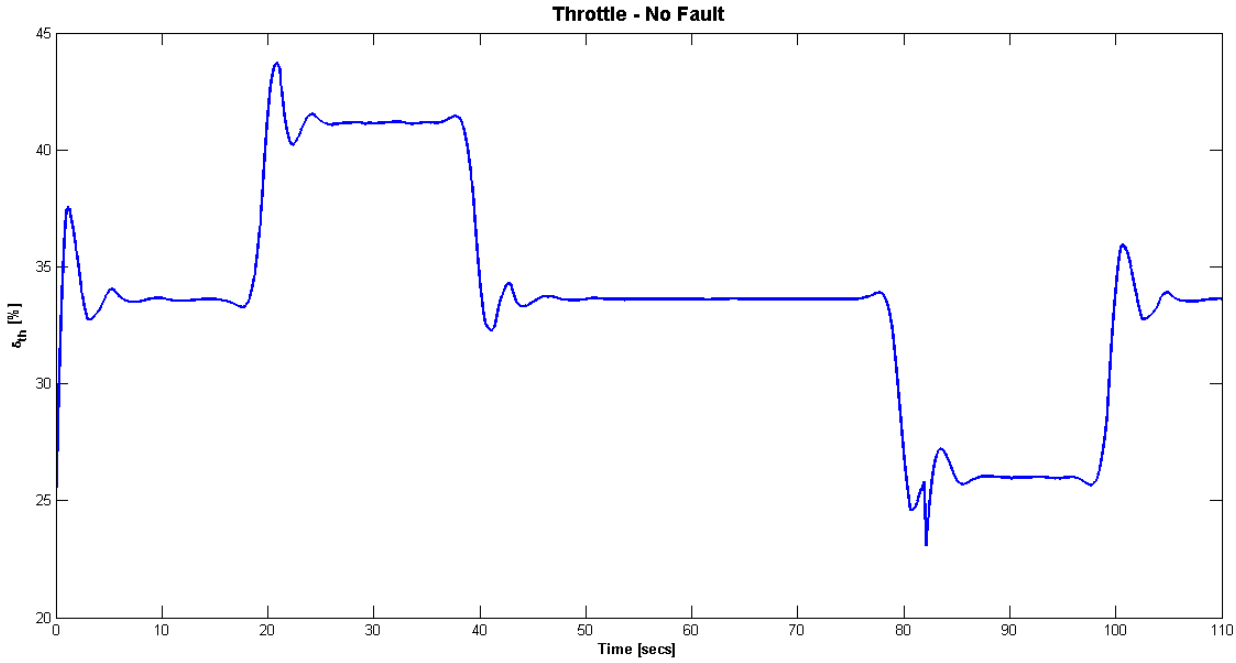

Figure 5.6: Throttle response - No Fault

It is also assumed that FDI information is available. This includes the time at which the throttle becomes stuck and and the position at which it is stuck. This information is used to update the constraint values of the NMPC controller. Providing the controller with the most accurate and up to date information enables it to make better use of the healthy actuators.

5.4.2 Numerical Results

To illustrate the concept of the FTC four different scenarios were set up:

Scenario 1: no fault case

Scenario 2: throttle stuck at 70% during entire flight,

Scenario 3: throttle stuck at 50% during entire flight,

Scenario 4: throttle stuck at 35% during entire flight,

Scenario 5: throttle stuck at 30% during entire flight,

Scenario 6: throttle stuck at 20% during entire flight,

Scenario 7: throttle stuck at 20% 80 secs into flight.

All plots given are of the optimal solution to illustrate proof of concept.

5.4.2.1 True Airspeed

The aircraft was required to maintain a $50m/s$ true airspeed. The plots given in figure 5.7 show the aircraft true airspeed for each of the scenarios. When the aircraft is fault free it is able to fly at the demanded true airspeed. However, when the throttle is stuck at 70% or even 50% there is too much power continually being provided to the aircraft resulting in large airspeed response. When the throttle is stuck at 35% the aircraft is able to maintain the demanded V_T for only a short period of time, at the beginning of the flight mission. However at 30% throttle the maximum deviation from the demanded airspeed is approximately $5m/s$ at any given time. When the throttle drops below 30% the aircraft is unable to maintain the true airspeed which drops to approximately between $35m/s$ and $30m/s$. The plot for scenario 7 shows that once the fault occurs at 80 secs the true airspeed immediately begins to drop, as expected. One main point to note is that the NMPC controller ensures that the stall speed is never reached. This is because the controller is designed to operate at 1.3 times the stall speed at a minimum.

5.4.2.2 Vertical Speed

The vertical speed response of the aircraft is shown in figure 5.8. The plots show the aircraft response along with the constraints (in red) placed on the vertical speed. For high values of throttle (70% and 50%) the vertical speed is continuously bouncing between the constraints in an attempt to maintain the true airspeed demand. For the case when the throttle is stuck at 35% the vertical speed profile is seen to be similar to the no fault case, except in the descent phase. During this phase, when the aircraft is descending and gaining speed, the vertical speed response can be seen to continuously move between the constraints to regulate the speed. For throttle values less than 30% there is insufficient power to maintain a climb hence the vertical speed is seen to operate at the lower constraint or at zero. The plot for scenario 7 shows that once the fault occurs at 80 secs the vertical speed moves between the constraints, working hard to maintain the true airspeed.

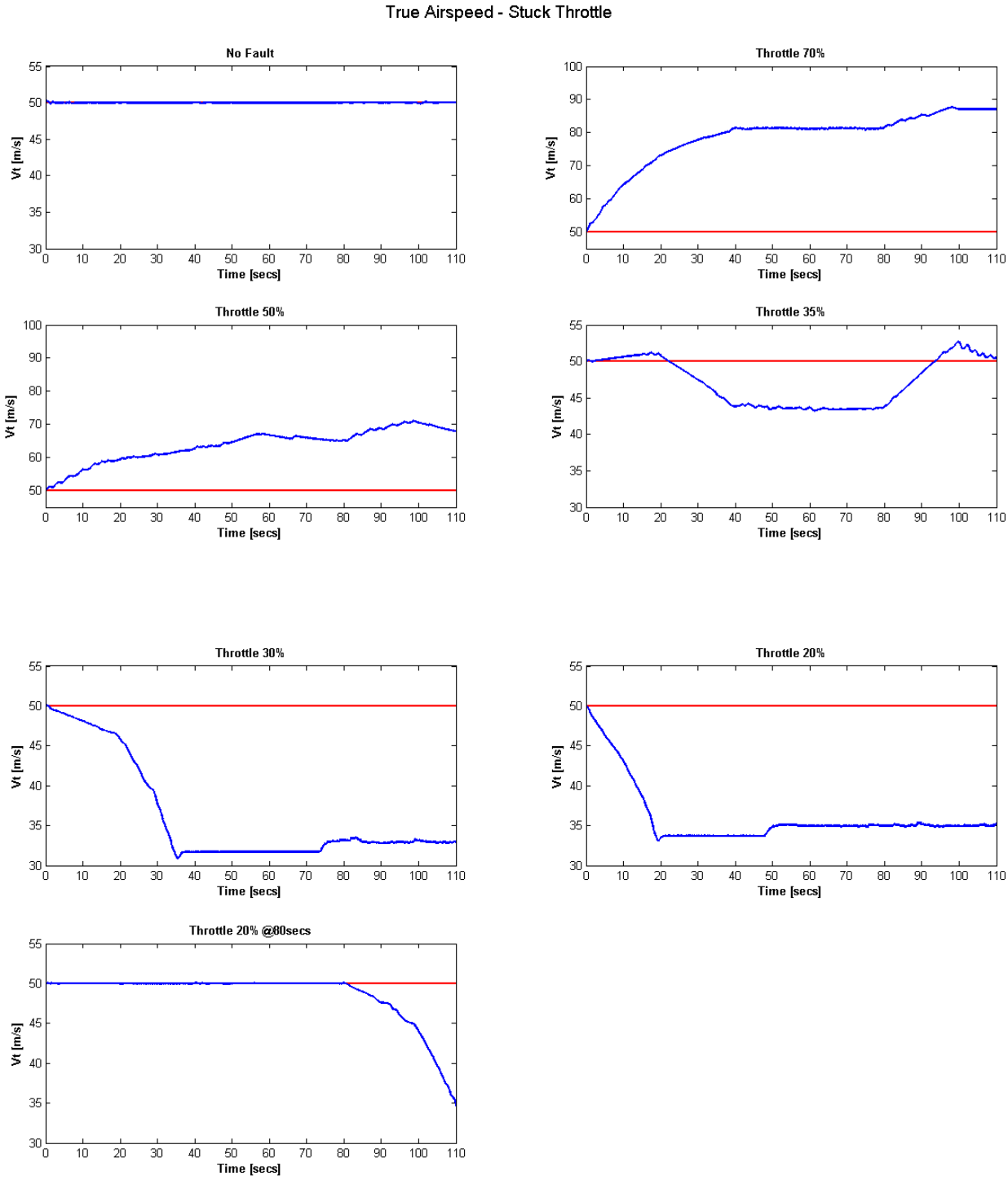

Figure 5.7: Stuck Throttle - True Airspeed Response, reference (red line), aircraft response (blue line)

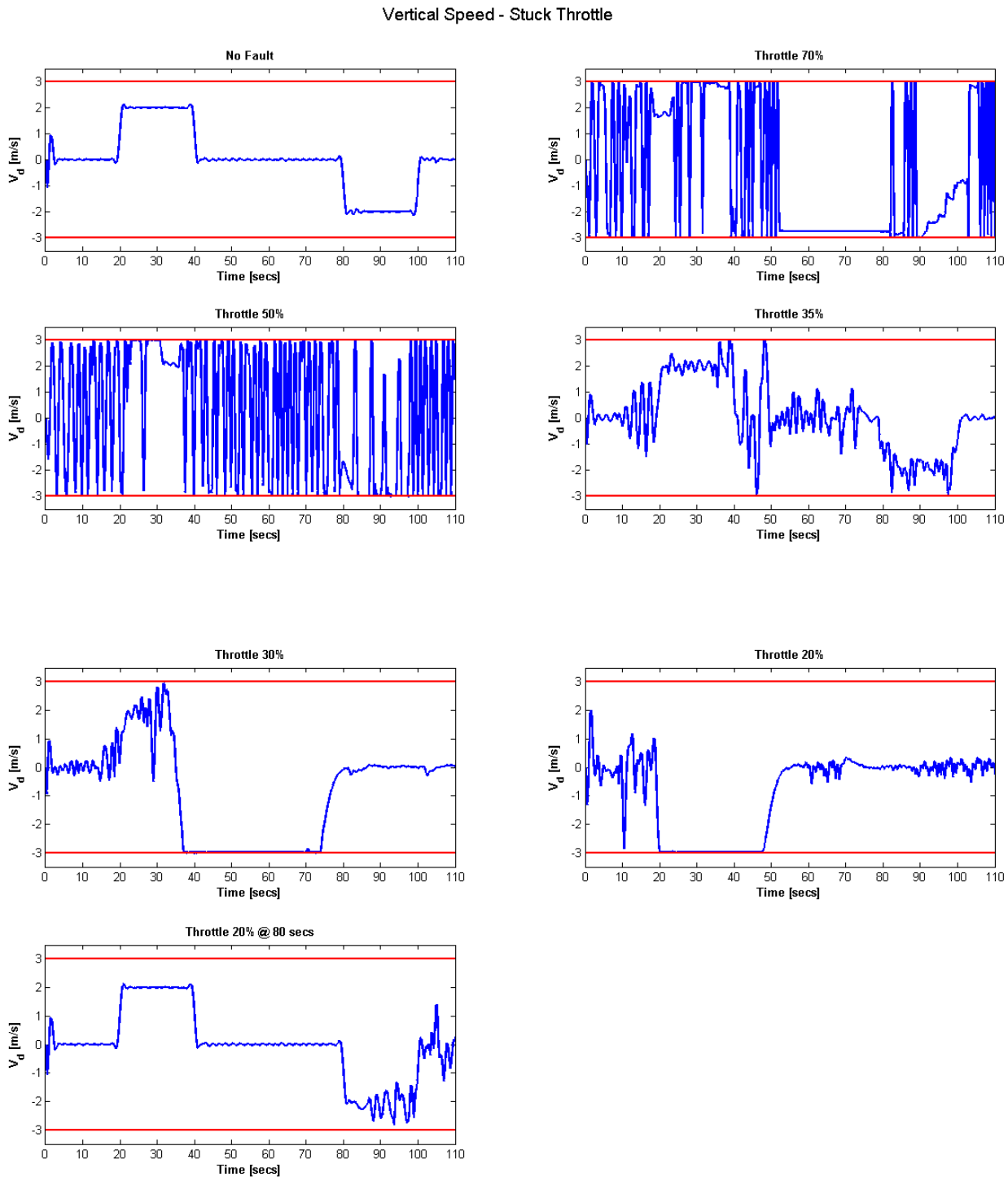

Figure 5.8: Stuck Throttle - Vertical Speed Response, constraints (red lines), aircraft response (blue)

5.4.2.3 Elevator Activity

In regards to fault tolerance, the elevator activity is of the most interest. If the throttle is stuck the elevator provides a level of redundancy to maintain the aircraft speed. Figure 5.9 shows plots of elevator activity for the different scenarios. The plots clearly show that any change in throttle increases the elevator activity when compared to the no fault case. The elevator activity increases in an attempt to regulate the airspeed of the aircraft. In the case of the high throttle values (70% and 50%) the elevator activity is the highest because a higher level of power is continually being provided to the aircraft exceeding the amount required to fly at the demanded speed. Hence the elevator constantly jumps between the constraints in an attempt to compensate for the excess power. For the 30% stuck throttle case the elevator activity does increase compared to the no fault case; however 35% throttle is closer to the amount required to maintain the given height profile (refer to figure 5.6), hence the elevator does not need to work as hard compared to the 70% and 50% cases. For the lower throttle values activity increases during the climb and descent phases. In the climb phase of the mission there is not enough power available to the aircraft, so it compensates by erratically deflecting the elevator. During the descent phase however there is too much power; to regulate this and to stay within the velocity constraints the activity again increases. The last scenario shows that at the fault occurrence time of 80 secs the elevator increases activity to compensate for the faulty throttle.

It is difficult to assess whether the rate constraints on the elevator are respected over the entire flight time. Hence the elevator activity for the first 10 seconds of flight for the elevator activity is shown in figure 5.10. These plots zoom in on the elevator activity and show that the rate constraints of 200 deg / sec are respected.

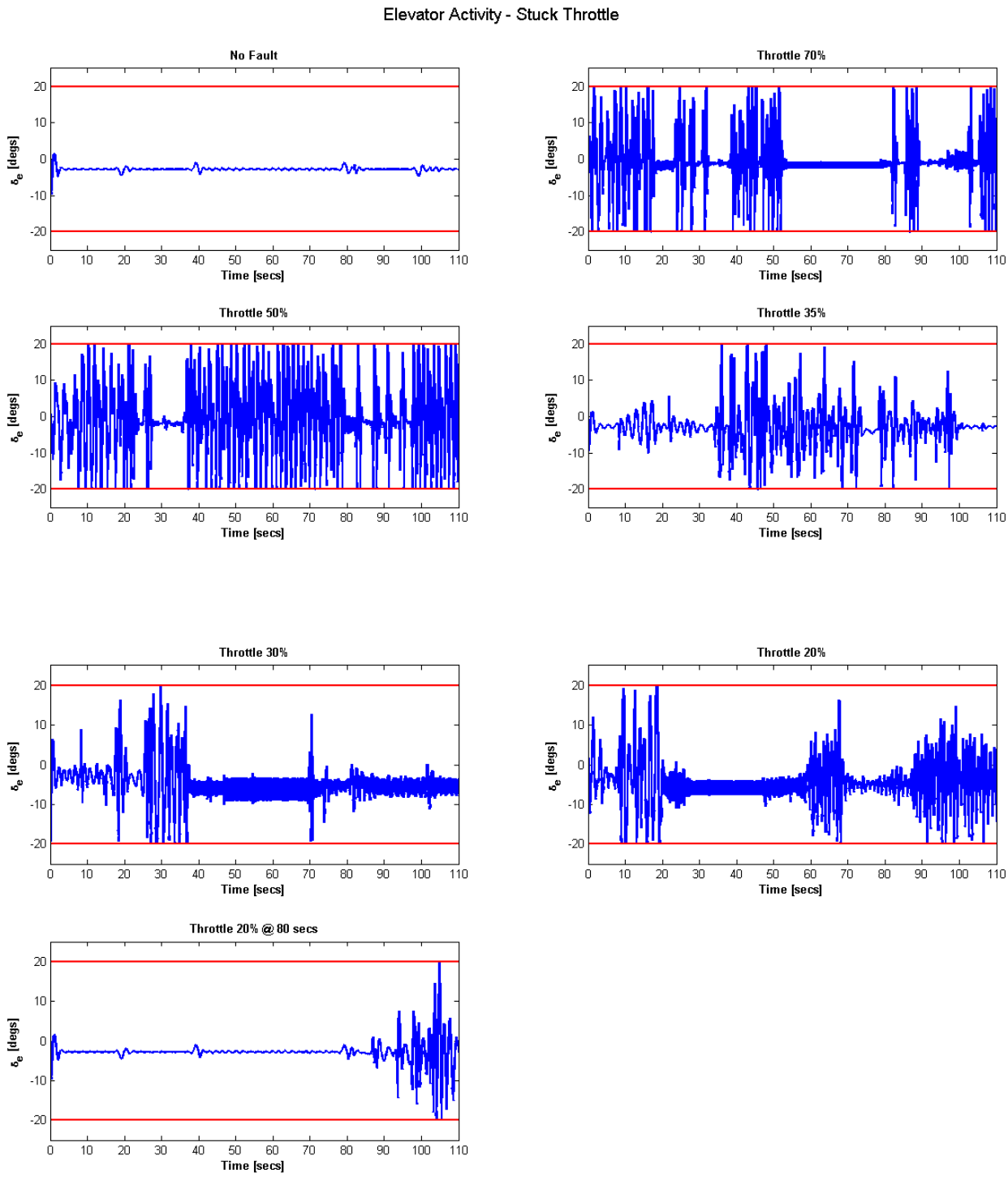

Figure 5.9: Stuck Throttle - Elevator Activity, constraints (red lines), aircraft response (blue)

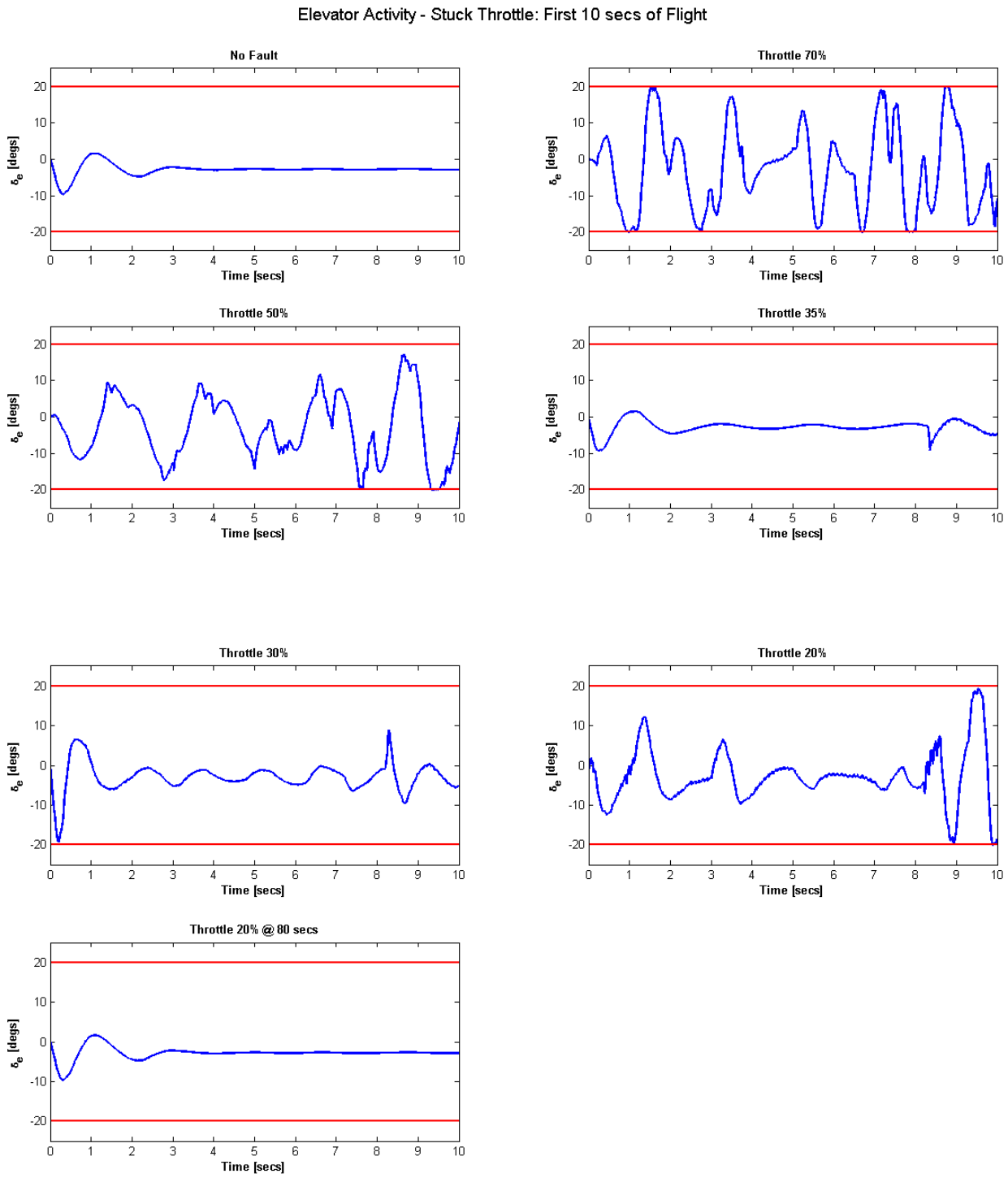

Figure 5.10: Stuck Throttle - Elevator Activity: First 10 secs of Flight, constraints (red lines), aircraft response (blue)

5.4.2.4 Pith Angle and Pitch Rate

The figures given in 5.11 and 5.12 show the plots of pitch angle and pitch rate respectively. The plots show that for the excessive power cases where the throttle is stuck at 70% and 50% the pitch rate and pitch angle continuously oscillate to maintain the speed of the aircraft. For the cases where little power is available the controller tries to maintain the pitch angle and pitch rate at a relatively constant rate forcing the the aircraft to glide down to safety.

5.4.2.5 Angle of Attack

Another pertinent state which needs to be analysed is the Angle of Attack (AoA). The model used is valid only for $\alpha \leq 15$ degrees and above this the equations of motion break down. The non-dimensional force and moment coefficients valid for $\alpha \geq 15$ degrees were provided by NASA however I chose to limit the research to one model rather than multiple models. For future work constraints should be placed on α or multiple models added. The plots given in figure 5.13 show the angle of attack during flight and it can be seen that at all times during the flight α remains below 15 degrees for any given scenario.

5.4.2.6 g-Force, Body Acceleration

The g-forces experienced by the aircraft during all scenarios are given in figure 5.14. It can be seen that the maximum g-force experienced during flight for any given scenario is approximately 1.5g, except for the 70% throttle case. This was to be expected as there is too much power available to the aircraft. In this case the maximum g-force is 3g's. The g-force is an important state to consider when assessing the possible structural damage that the aircraft could experience.

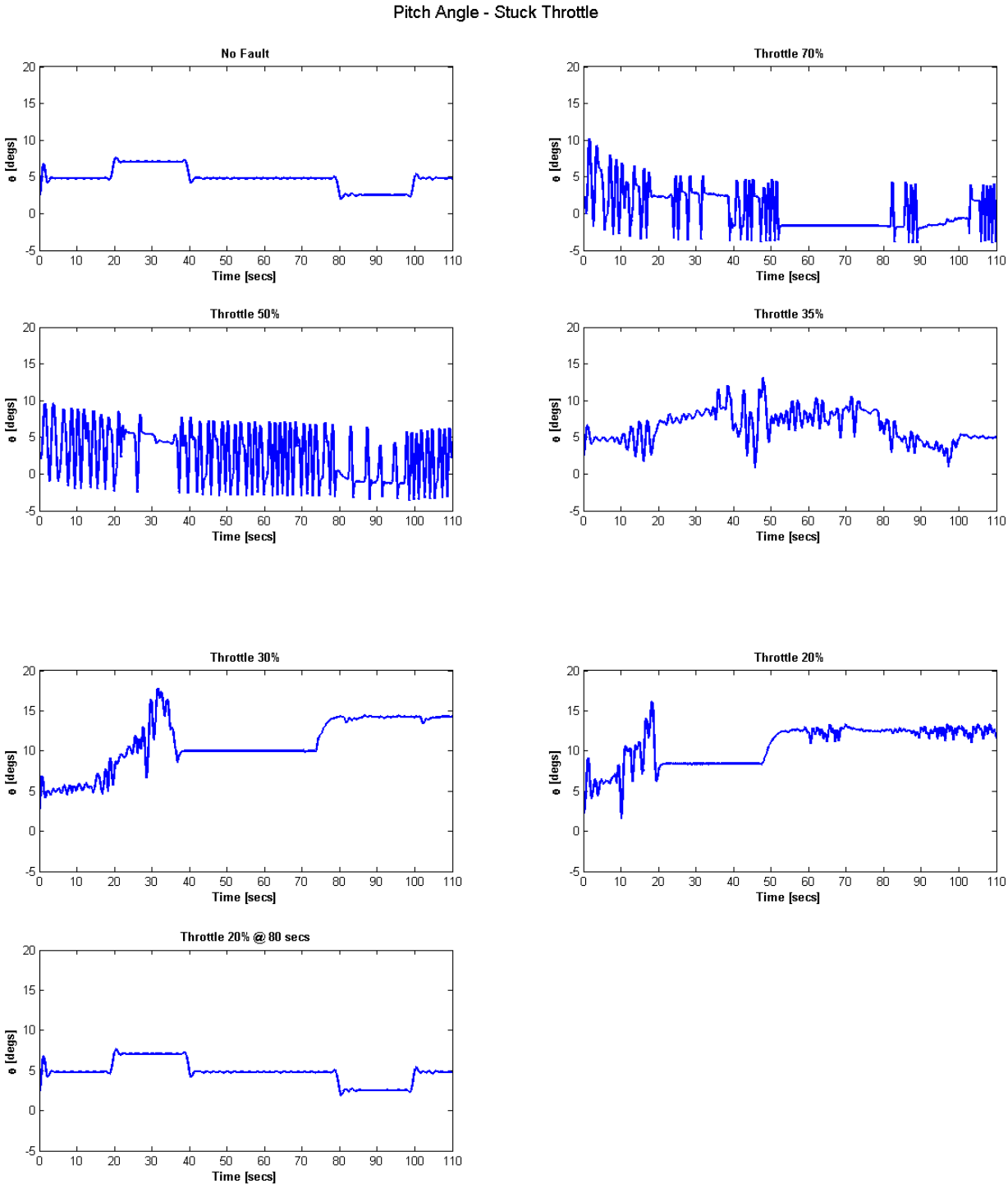

Figure 5.11: Stuck Throttle - Pitch Angle, θ

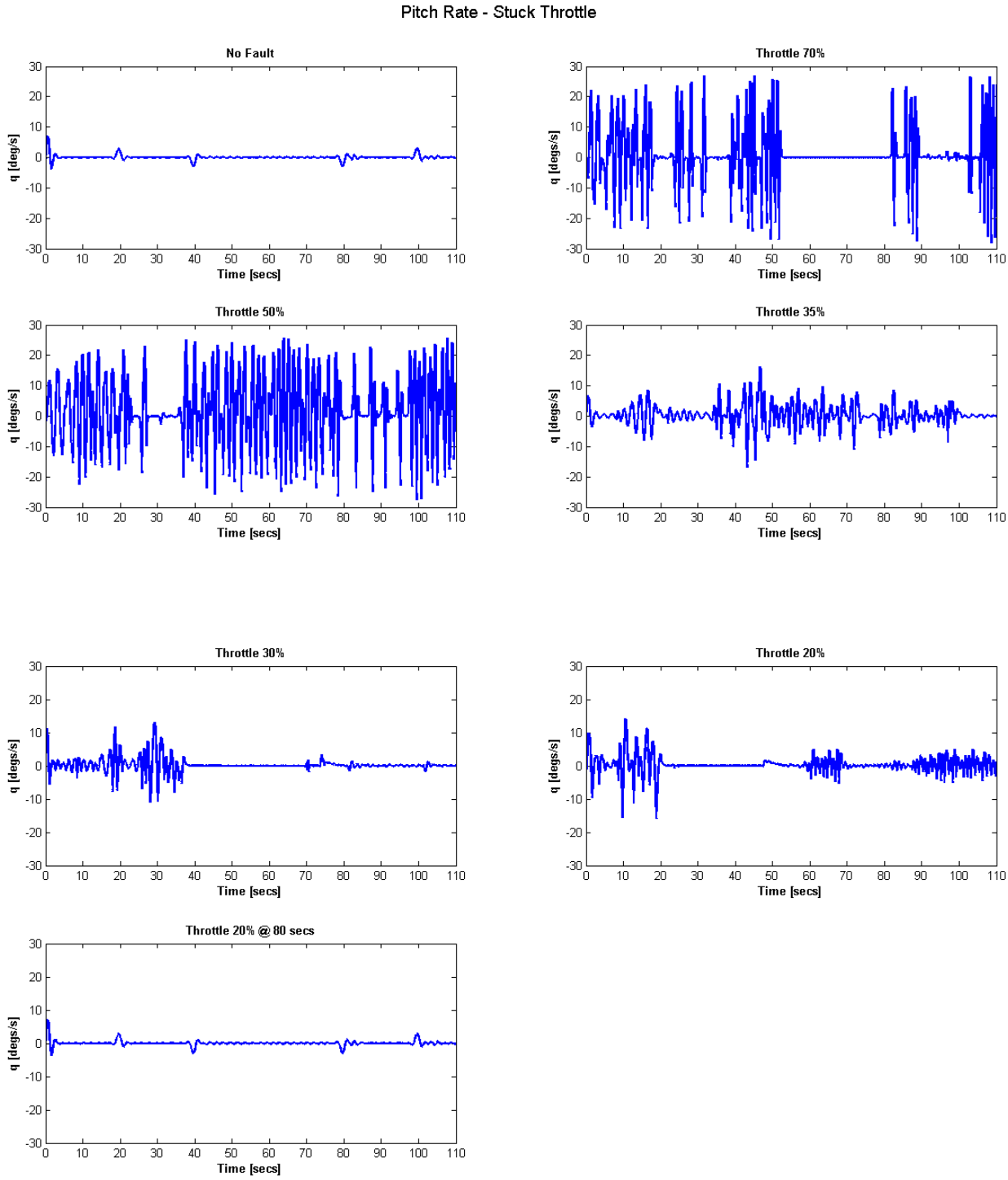

Figure 5.12: Stuck Throttle - Pitch Rate, q

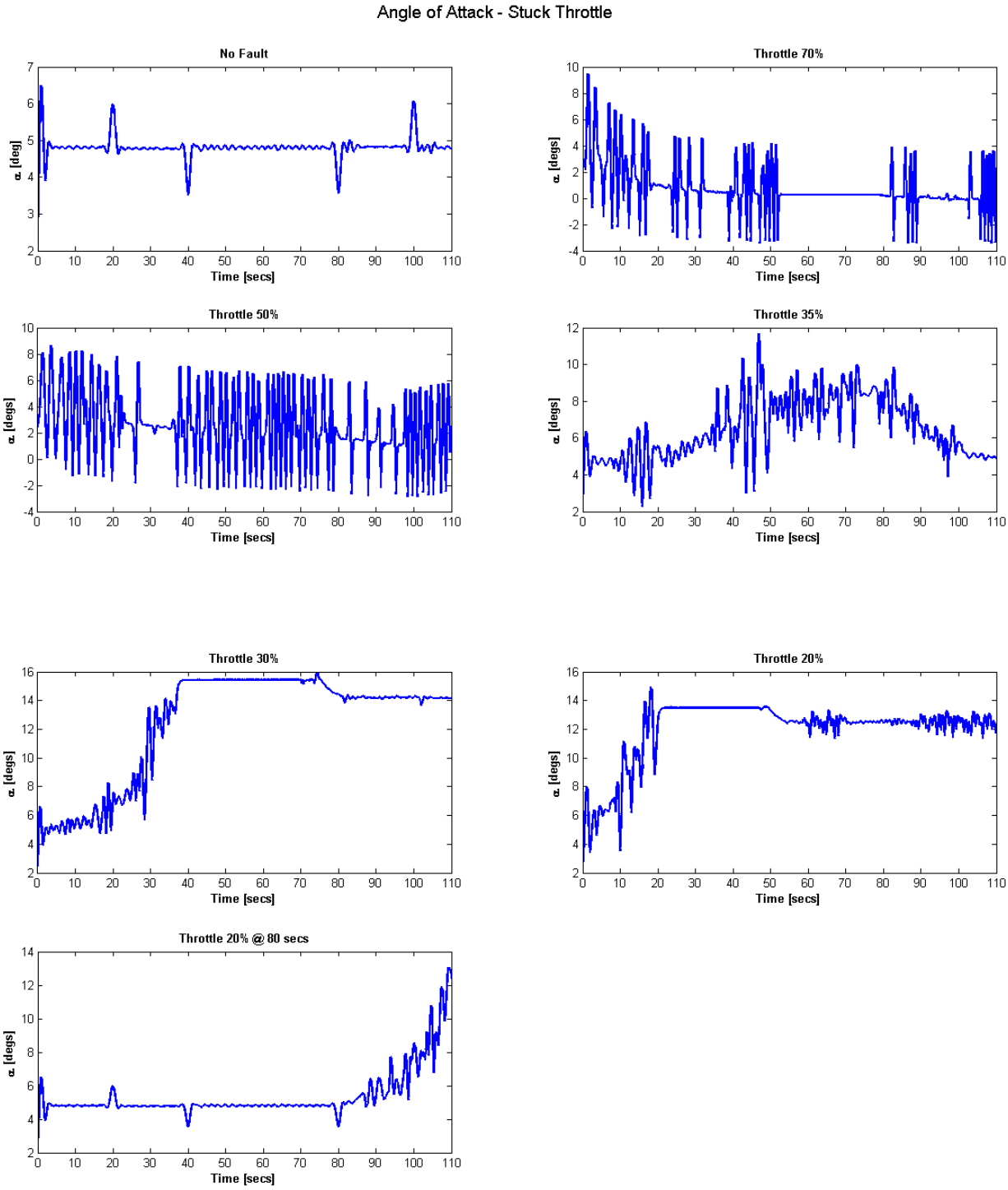

Figure 5.13: Stuck Throttle - Angle of Attack

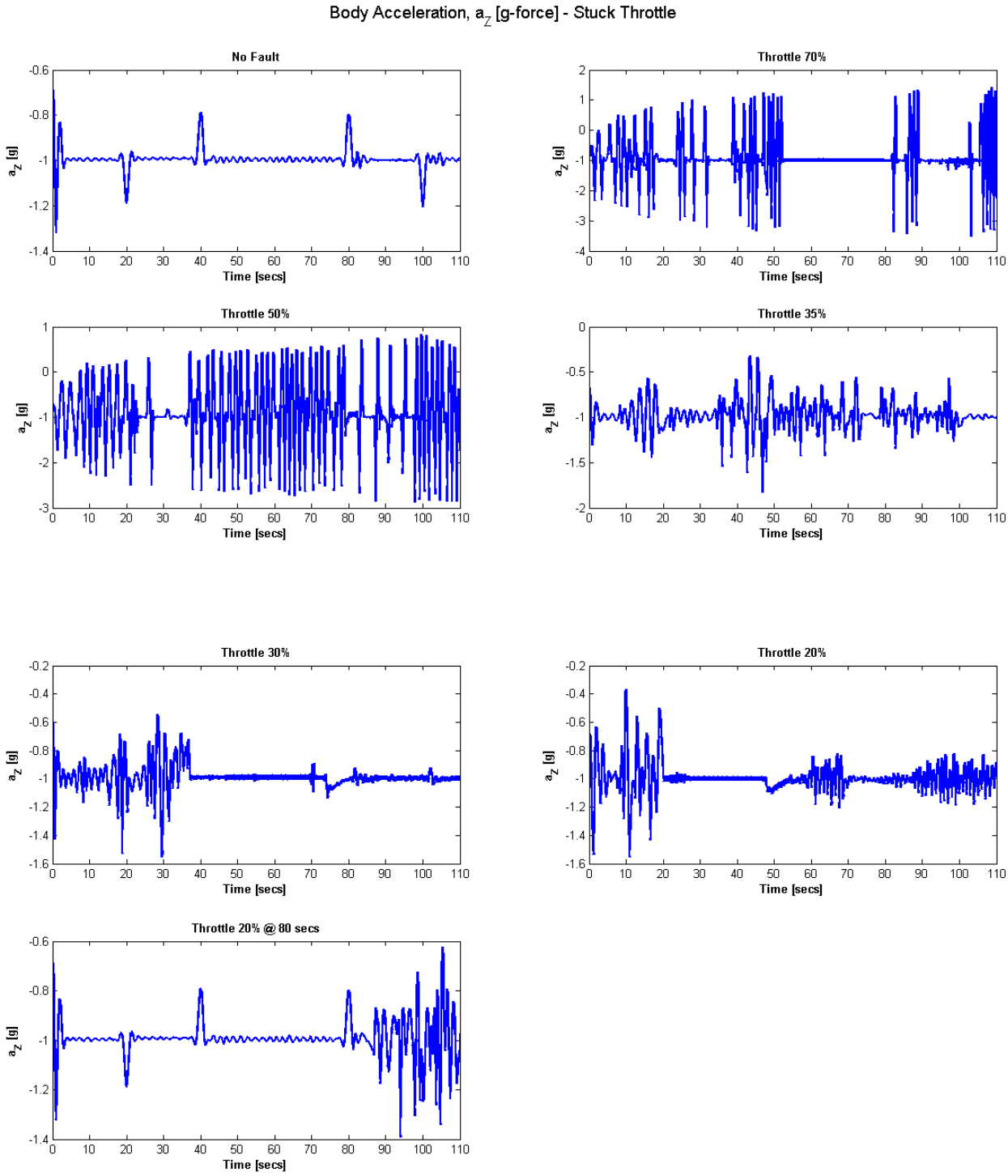

Figure 5.14: Stuck Throttle - g Force (body acceleration, a_z

5.4.2.7 Height Profile

The trajectory flown by the aircraft during the different scenarios is given in figure 5.15. The no fault case, as expected, follows the reference height profile perfectly. The 35% case is also able to closely maintain the profile. In the 70% and 50% cases the aircraft continually tries to regulate the airspeed to compensate for the excess power. The solutions produced by both scenarios show the aircraft overshooting followed by an undershoot, so the solution oscillates around the reference. When the throttle becomes stuck at 30% the aircraft begins the climb phase of the mission but is only able to continue climbing for 20 secs before it begins gliding towards the ground. In the 20% stuck throttle case the aircraft completes the straight and level phase of the mission but does not have enough power to begin climbing, and descends to the ground. The final scenario shows that the elevator is able to compensate for the stuck throttle in mid-flight and successfully finish the mission.

5.4.3 Findings

The results given in this section illustrate that the nonlinear NMPC controller is a viable solution for fault tolerant flight control. This is evident from the results which show that in the event of a stuck throttle the controller is able to manipulate the movement of the elevator to compensate for the fault.

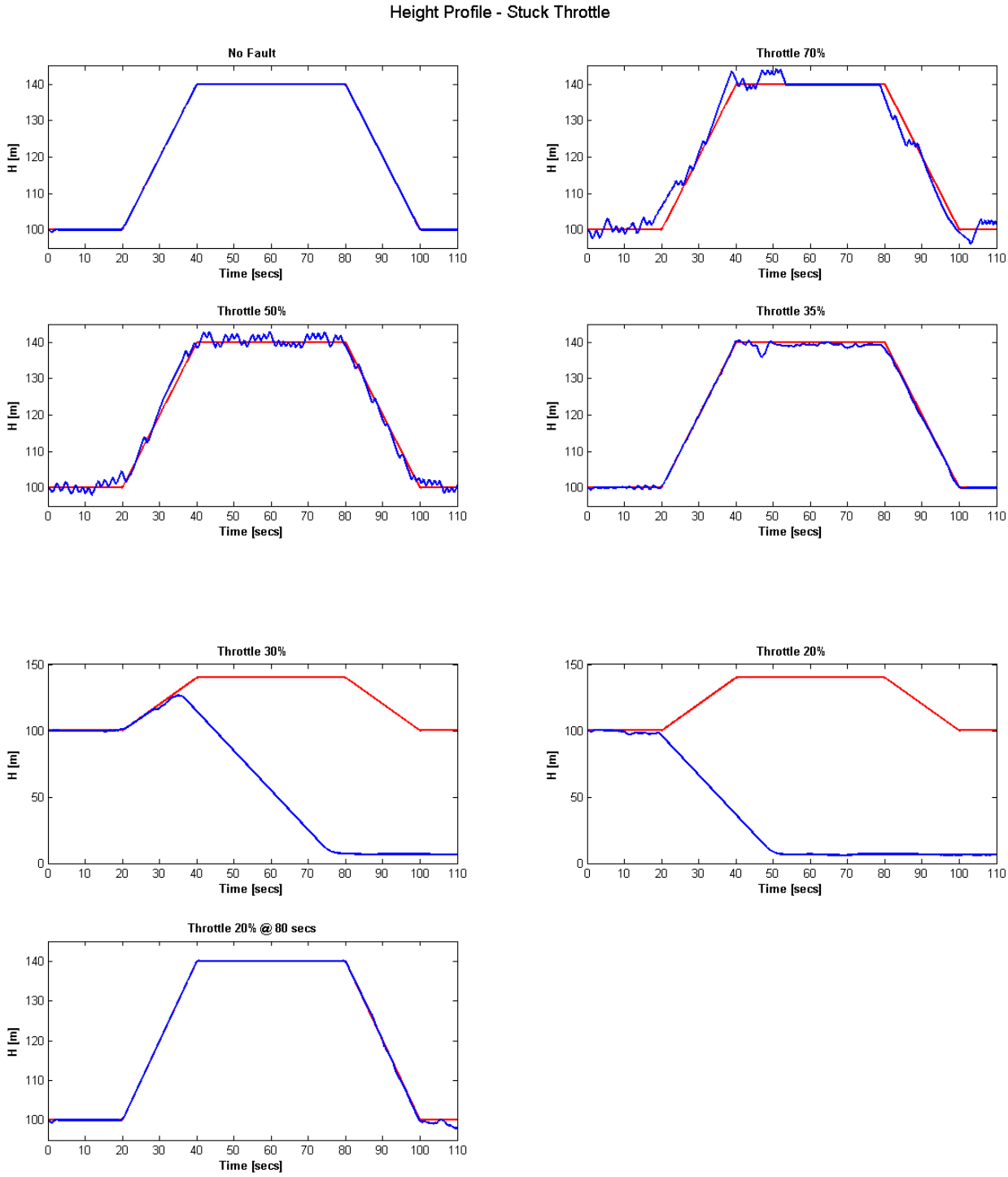

Figure 5.15: Stuck Throttle - Height Profile, reference (red lines), aircraft response (blue)

In the next section the full 6DoF model of an aircraft will be employed, and an NMPC controller implemented.

5.5 Six Degree of Freedom Aircraft Model with Fault Tolerant Control

Parts of this section have appeared as [72].

In section 5.3 the full 6DoF aircraft model was introduced and the longitudinal motion of the aircraft was investigated for fault tolerant control in section 5.4. In this section the full 6DoF model is used and an NMPC controller is designed to control the aircraft and handle any unforeseen faults.

5.5.1 6DoF Controller Design

The system given in 5.2 is modelled in this section with the full 6DoF aircraft model. Again the plant model doubles as the navigation model and the guidance system has been provided by Williams [127]. The system was modelled in Simulink and the Dryden Wind Turbulence model from the Aerospace Blockset [66] was used to model wind and turbulence. The guidance subsystem is supplied with a series of way points and it provides the controller with angular rate information, ie. it calculates the angular rates required to maintain the reference path and are referred to as angular rate demands. The inner loops consists of two controllers, an NMPC controller to control angular rates and a speed control loop which is a simple PI controller to maintain a desired speed. An integrator is also implemented to calculate the integrated errors in the angular rates.

A pseudospectral discretisation method is used and the state vector of the NMPC is given by:

$$\mathbf{x}_{\text{nmpc}} = [p,\ q,\ r,\ I_p,\ I_q,\ I_r,\ \delta_e,\ \delta_a,\ \delta_r,\ \Delta\delta_e,\ \Delta\delta_a,\ \Delta\delta_r]^\intercal\,, \tag{5.43}$$

where p, q and r are the roll rate, pitch rate and yaw rate respectively, I_p, I_q and I_r are respectively, the integrated errors in p, q and r used to minimise the steady state errors, and δ_e, δ_a and δ_r are the elevator, aileron and rudder deflections respectively.

The prediction model of the nonlinear MPC controller is as follows:

$$\dot{p} = (c_1\, r + c_2\, p + c_4\, h_{eng})\, q + \bar{q}\, S\, b\, (c_3\, C_l + c_4\, C_n)\,, \tag{5.44}$$

$$\dot{q} = (c_5\, p - c_7\, h_{eng})\, r - c_6\, \left(p^2 - r^2\right) + \bar{q}\, S\, \bar{c}\, c_7\, C_m, \tag{5.45}$$

$$\dot{r} = (c_8\, p - c_2\, r + c_9\, h_{eng})\, q + \bar{q}\, S\, b\, (c_4\, C_l + c_9\, C_n)\,, \tag{5.46}$$

$$\dot{I}_p = \hat{p} - p_{dem}, \tag{5.47}$$

$$\dot{I}_q = \hat{q} - q_{dem}, \tag{5.48}$$

$$\dot{I}_r = \hat{r} - r_{dem}, \tag{5.49}$$

where $\hat{p}$, $\hat{q}$ and $\hat{r}$ are the predicted angular rates and p_{dem}, q_{dem} and r_{dem} are the demanded angular rates. The remaining parameters given in the equations above are as defined in section 5.3.

The following control problem is solved at each time step:

$$J = \min_{\mathbf{x},\mathbf{u}} \frac{H_p}{2} \sum_{j=1}^{j=N+1} \left(\left\| \omega(j) - \omega_{\mathrm{dem}}(j) \right\|_{Q_\omega}^2 + \left\| \mathbf{I}(j) \right\|_{Q_I}^2 + \left\| \Delta\mathbf{u}(j) \right\|_{Q_u}^2 \right) w(j), \tag{5.50}$$

subject to

$$\left(\frac{t_f - t_0}{2} \right) \mathbf{D}_{j,k} \mathbf{x}\,(t) - \dot{\mathbf{x}} = 0, \tag{5.51}$$

$$\omega(t_0) - \omega_{\mathrm{dem}}(t_0) = 0, \tag{5.52}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{5.53}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}, \tag{5.54}$$

$$\Delta\mathbf{u}_{lb} \leq \Delta\mathbf{u} \leq \Delta\mathbf{u}_{ub}, \tag{5.55}$$

with the constraints on the states as in table 5.5.

Table 5.5: Constraints for 6DoF Motion

Variable	**Upper Constraint**	**Lower Constraint**
p	None	None
q	None	None
r	None	None
I_p	None	None
I_q	None	None
I_r	None	None
δ_A	20 deg	-20 deg
δ_E	20 deg	-20 deg
δ_R	20 deg	-20 deg
Δd_A	200 deg / sec	-200 deg / sec
Δd_E	200 deg / sec	-200 deg / sec
Δd_R	200 deg / sec	-200 deg / sec

A prediction window length of 1 second was used with 16 coincidence points. A 1 second window was deemed sufficient for the purposes of angular rate following as it is assumed that the angular rate demands are constant across the window length. This is a reasonable assumption because the angular rates do not change significantly after 1 second. The cost function weights are square matrices with the following diagonal values for each state: $Q_\omega = 1$, $Q_I = 0.001$ and $Q_u = 0.01$.

5.5.1.1 Fault Simulation

The concept behind the fault tolerant controller design for the 6DoF model is based on monitoring the control derivatives. The non-dimensional aerodynamic coefficients for the forces and moments given in section 5.3.1.2 are made up of a series of aerodynamic and control derivatives. For example the term $-6.54 \times 10^{-3}\delta_e$ in the pitching moment coefficient equation (5.30) represents the pitch control derivative, $C_{m_{\delta_e}}$, the contribution of the elevator control input on the pitching moment coefficient. The contributions made by α and β are known as the aerodynamic derivatives. In the example given ($C_{m_{\delta_e}} = -6.54 \times 10^{-3}$) the value -6.54×10^{-3} is specific to the given aircraft as are all the derivative values given in equations (5.26), (5.27), (5.28), (5.29), (5.30) and (5.31). For any aircraft these values are obtained via experimental testing or computational fluid dynamic techniques, the derivatives are affected by any physical change in

the control surface hence any change in a control derivative would indicate a fault.

For the simulation results given in subsection 5.5.2 the faults are simulated by reducing the efficiency of the control surface. The primary role of an elevator is to provide pitch control, so its largest contribution is on the pitching moment, and therefore a change in the $C_{m_{\delta_e}}$ derivative would indicate an elevator fault. The aileron contributes primarily to the rolling moment C_l and the control derivative associated with the aileron from equation (5.29) is $C_{l_{\delta_a}} = 6.1 \times 10^{-4}$. Finally the rudder has the biggest impact on the yawing moment, C_n, and the associated control derivative from equation (5.31) is $C_{n_{\delta_r}} = -9.0 \times 10^{-4}$. To simulate a fault in a control surface the respective control derivative is reduced.

5.5.2 Numerical Results

To investigate the effectiveness of the NMPC controller design as a fault tolerant controller the aircraft was required to fly the trajectory given in figure 5.16.

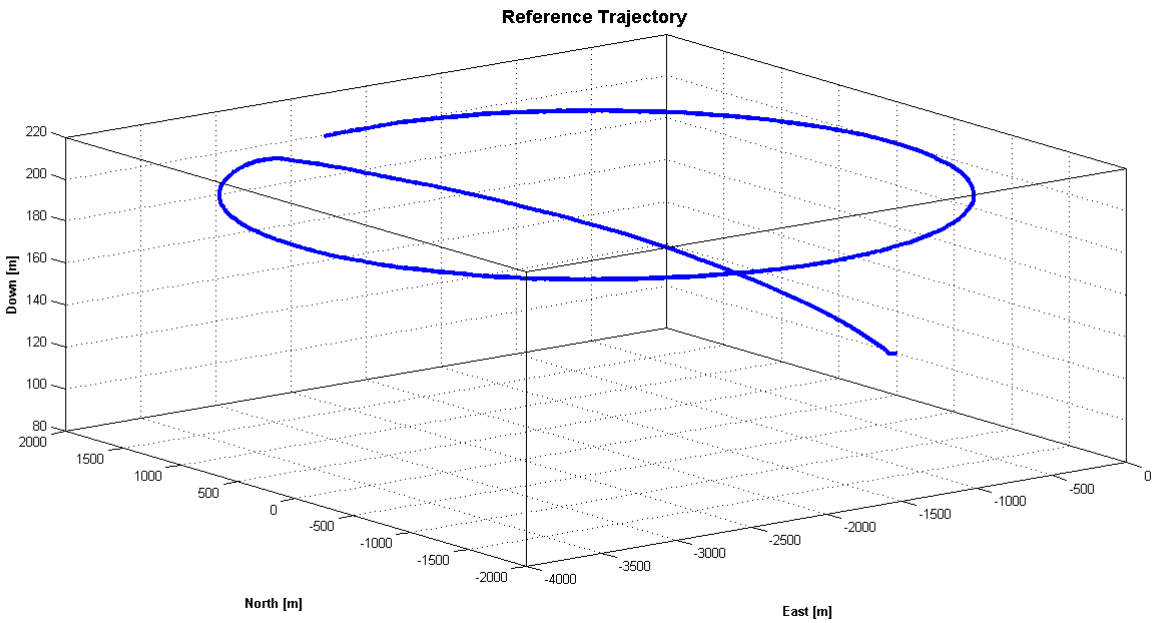

Figure 5.16: 6DoF Reference Trajectory

Three different scenarios were set up:

Scenario 1: faulty elevator: 70% reduction in efficiency 20 seconds into flight,

Scenario 2: faulty aileron: 80% reduction in efficiency 20 seconds into flight,

Scenario 3: faulty rudder: 60% reduction in efficiency 20 seconds into flight,

Each scenario is run with and without FDI information. FDI information is assumed and provides details on the time of fault and the efficiency of the control surface.

5.5.2.1 Scenario 1: Faulty Elevator

Figure 5.17 presents the plots for the control surface activity given an elevator with 70% reduction in efficiency. The plots show that without any knowledge of the fault the activity in the elevator decreases after 20 seconds and there is very little change in the aileron and rudder activity once the fault occurs. When FDI information is provided however the knowledge of the fault prompts the control surfaces to work harder to compensate for the fault. This is seen in all three control surfaces which at various times during the flight are all operating at the constraints.

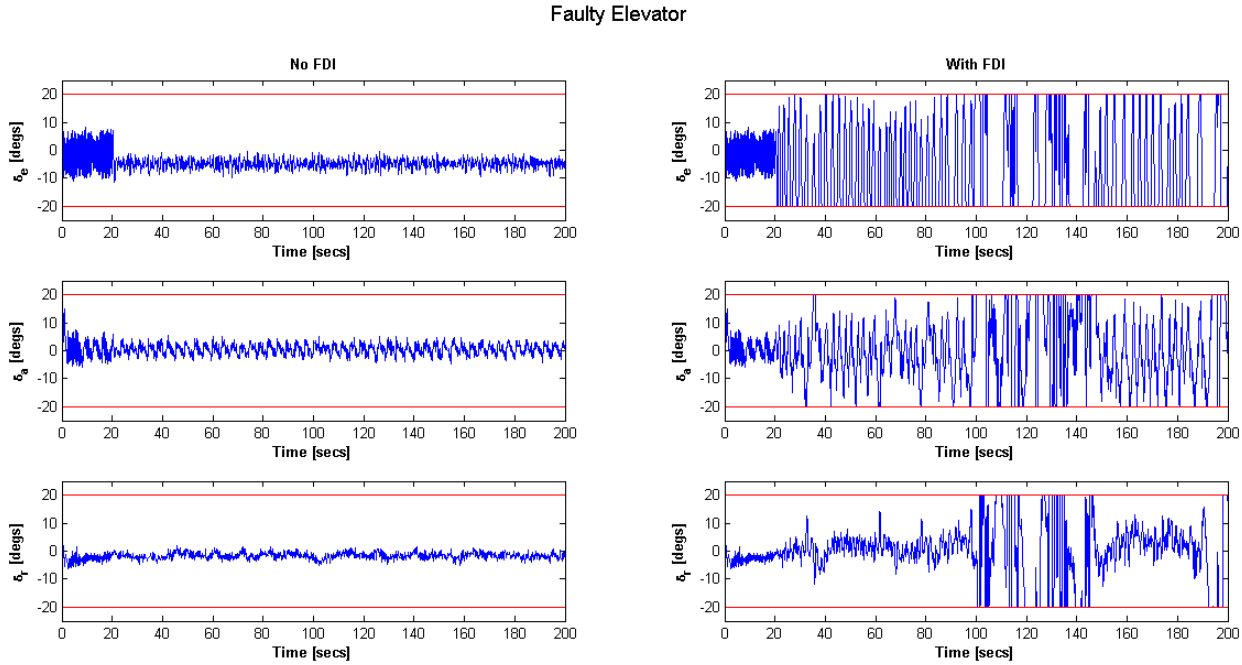

Figure 5.17: Faulty Elevator: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information

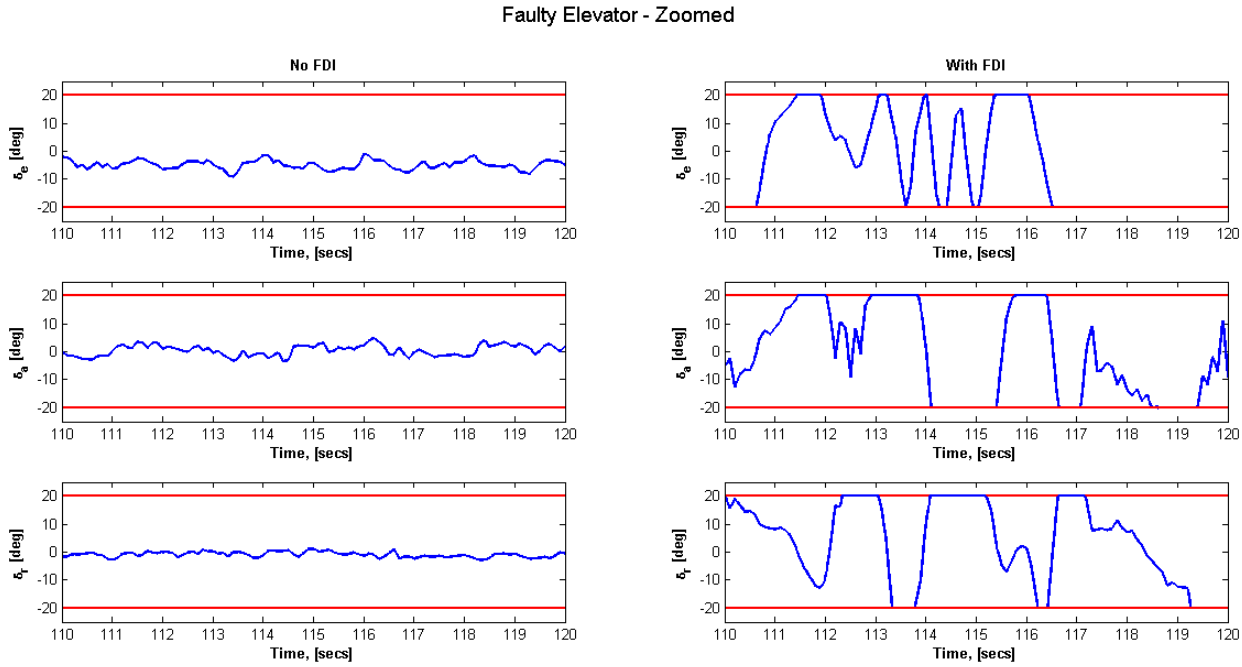

Figure 5.18: Faulty Elevator: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information - Zoomed

Figure 5.18 shows the elevator activity between 110 secs and 120 secs. During this time large amounts of oscillations can be observed (figure 5.17) however by zooming in at this time the plot shows that the rate constraints on the control surfaces are being respected. This is to be expected as the actuator dynamics have been modelled in the controller as well as in the plant model.

The angular rate plots are shown in figure 5.19. A fault in the elevator directly affects the pitch rate q, and without any FDI information the controller is unable to meet the pitch rate demands however the roll rate and yaw rate demands are followed very closely. With knowledge of the faults there is an increase in the demanded angular rates and the controller shows a significant improvement in performance in being able to follow the demanded rates.

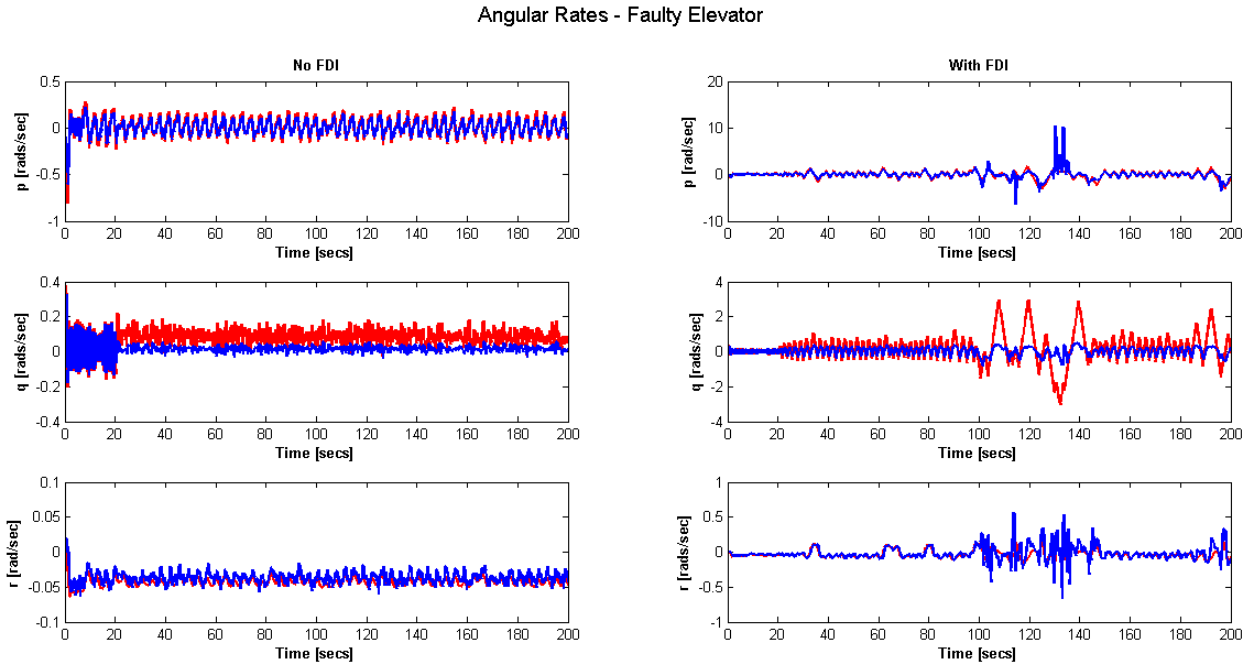

Figure 5.19: Faulty Elevator: Angular Rates, demanded (red), actual (blue). Left column: no FDI information, Right column: with FDI information

Plots of AoA and g-force are give in 5.20 and 5.21 respectively. The AoA plots shows that the 15 deg upper limit on AoA was never reached hence the equations of motion in the process model were valid throughout the flight. The maximum g-force reached was only 1.5g which is very unlikely to cause structural damage.

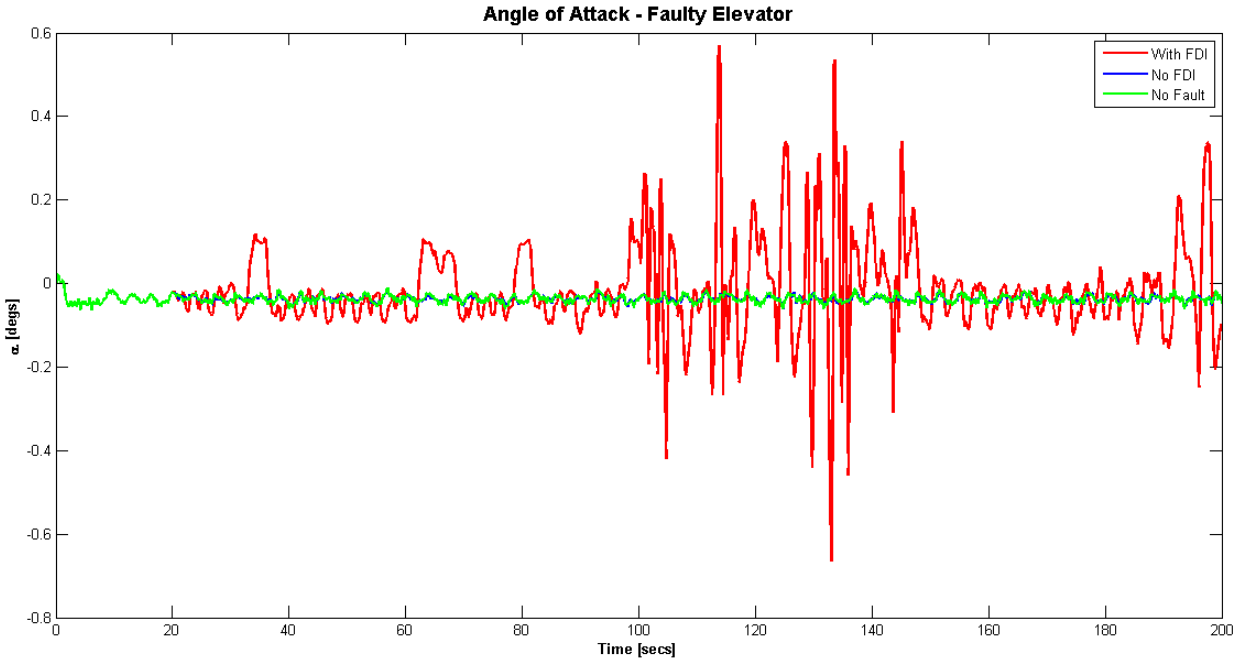

Figure 5.20: Faulty Elevator: Angle of Attack

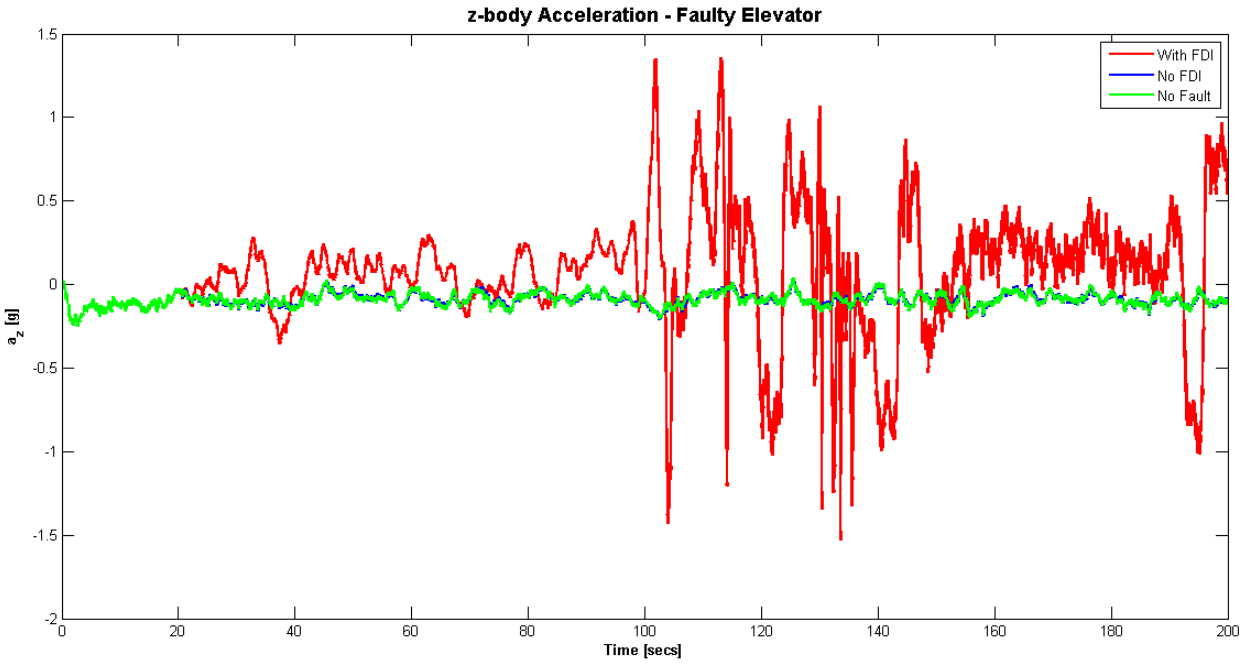

Figure 5.21: Faulty Elevator: g-Force, Body Acceleration a_Z

The trajectories flown by the aircraft with a faulty elevator, with and without FDI information, are provided in 5.22. The results show that in the absence of FDI information the aircraft successfully flies the trajectory, however providing FDI information caused the solution to diverge.

This result shows that the controller behaved exactly as expected. The controller has been designed to maintain the angular rate demands not the reference trajectory. The angular rate plots show that with the FDI information there is an increase in performance of the controller in terms of tracking the angular rate demands. The trajectory plots show that the solution produced with FDI information causes the aircraft to drop below ground level which is physically impossible. This is a result of not applying constraints on the aircraft position vector. Hence unless a parameter is explicitly penalised in the cost function and/or constraints placed upon the parameters the controller will use everything available to it to achieve what is being demanded of it. A zoomed in plot of the NED states are given in figure 5.23. The plot shows that upon zooming in the case with no FDI the controller does a an excellent job of following the no-fault situation however in the case where FDI information is available the solution diverges.

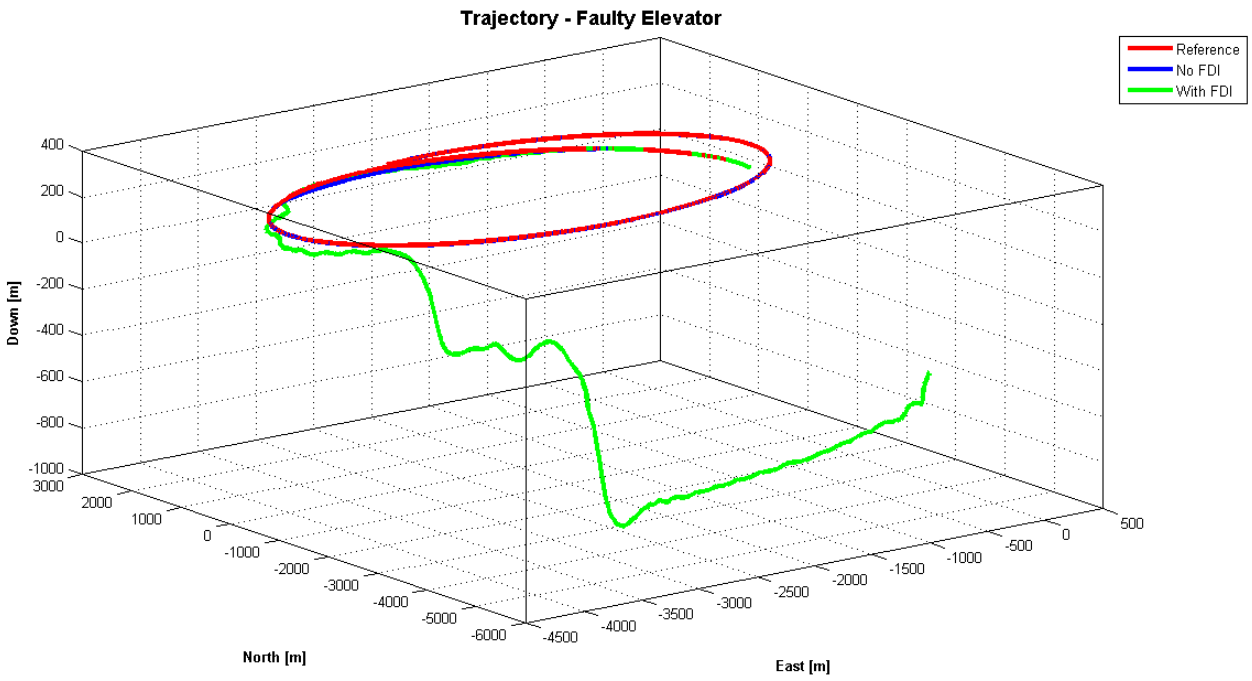

Figure 5.22: Faulty Elevator: 6DoF Trajectory

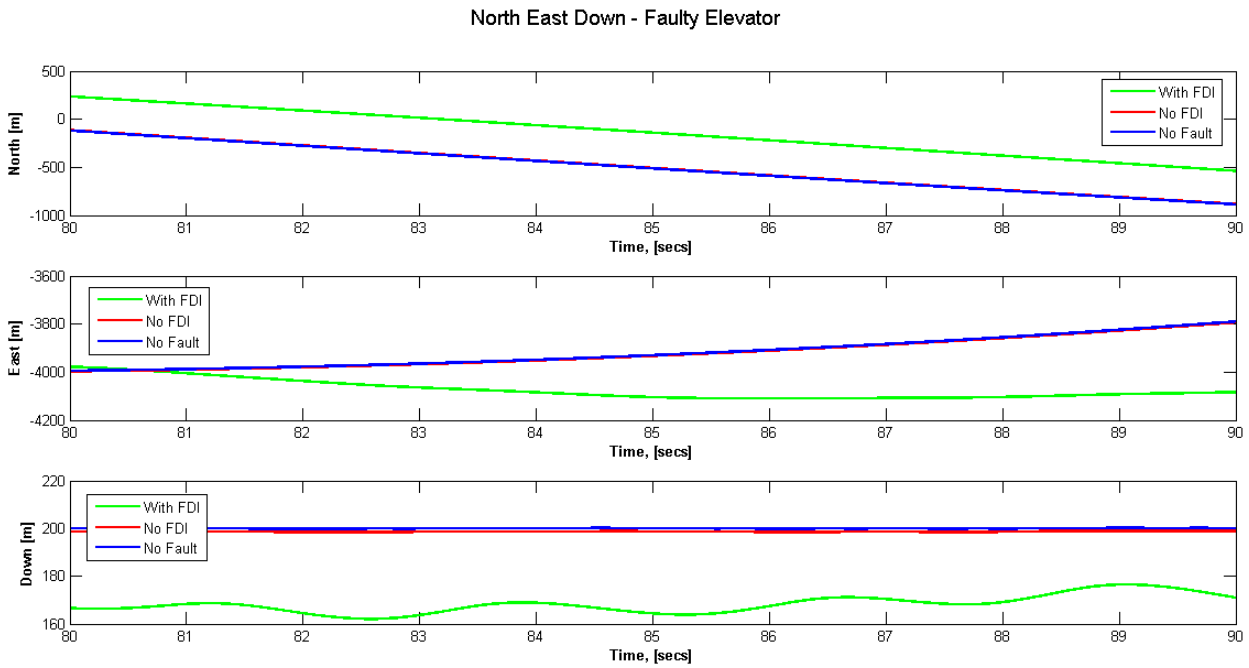

Figure 5.23: Faulty Elevator: NED

5.5.2.2 Scenario 2: Faulty Aileron

The plots given in figure 5.24 show control surface activity with an 80% reduction in aileron efficiency. The red lines show the constraints placed on the control surface and the blue the actual activity. The aileron and rudder are primarily used to control the lateral motion of the aircraft while the elevator controls the longitudinal motion. Thus there is very little change in the behaviour of the elevator when FDI information is provided compared to no FDI information. The rudder and aileron on the other hand increase their activity after the occurrence of the fault to compensate for the loss in efficiency and operate closer to the constraints.

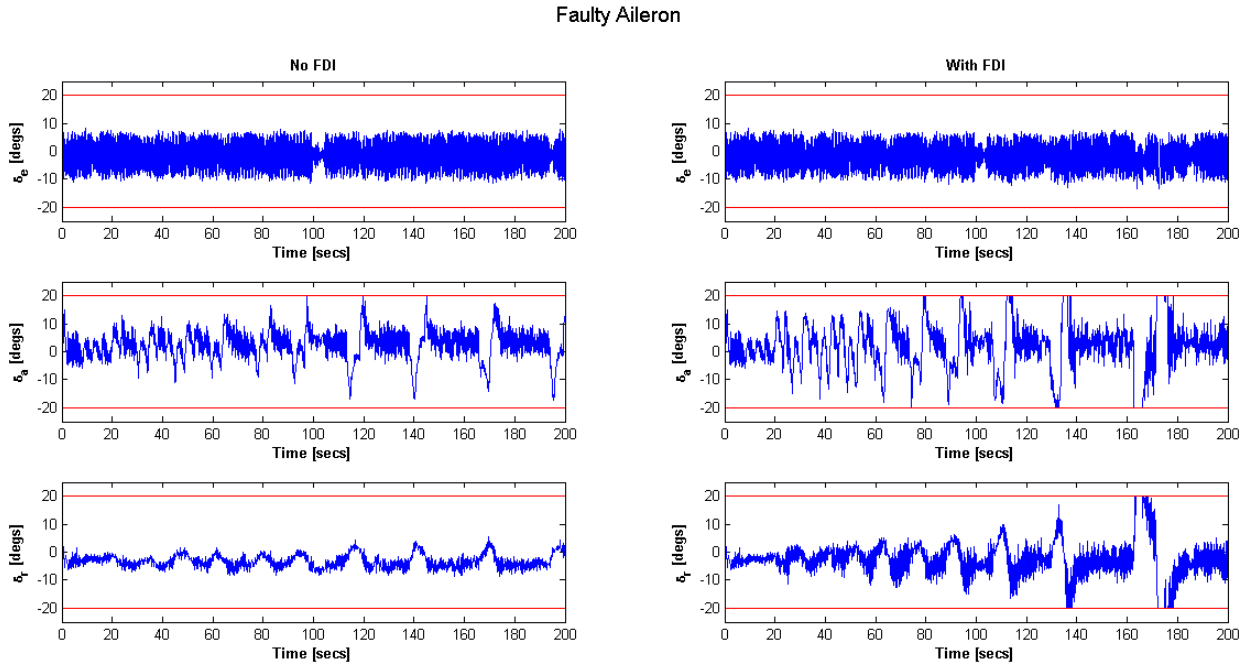

Figure 5.24: Faulty Aileron: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information

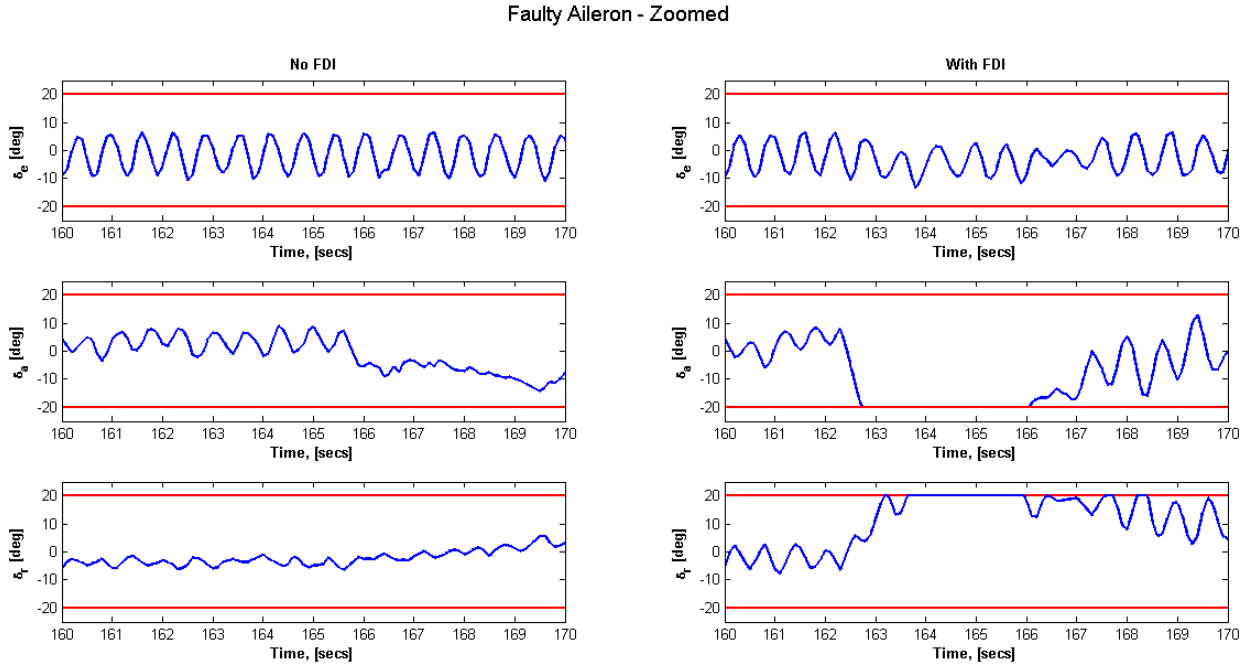

Figure 5.25: Faulty Aileron: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information - Zoomed

Figure 5.25 shows the elevator activity between 160 secs and 170 secs. During this time large

amounts of oscillations can be observed (figure 5.24) however by zooming in at this time the plot shows that the rate constraints on the control surfaces are being respected. This is to be expected as the actuator dynamics have been modelled in the controller as well as in the plant model.

Similar results are present in the angular rate plots of figure 5.26. There is little or no change in the pitch response of the aircraft once FDI information is provided compared when FDI is absent. In the case of no FDI the actual roll rate is lower than the demand however once information on the fault is provided tracking performance increases. This is also true for the yaw rate response. In the presence of an aileron fault roll and yaw rate demands increase to sustain lateral motion.

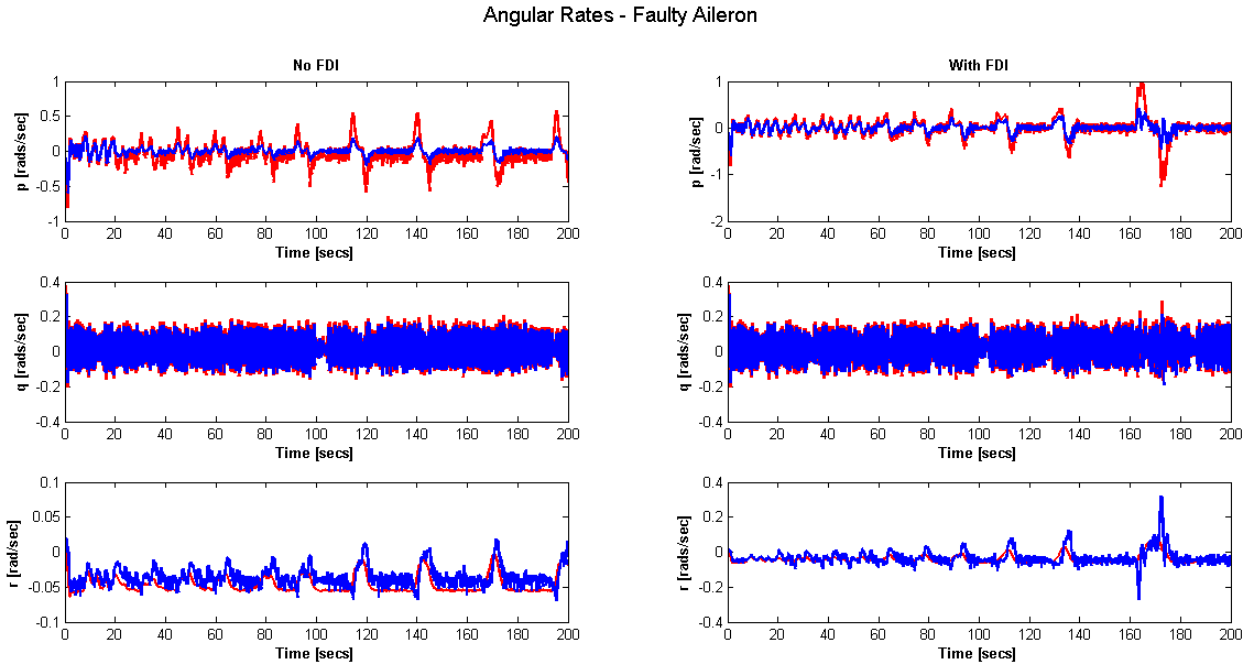

Figure 5.26: Faulty Aileron: Angular Rates, demanded (red), actual (blue). Left column: no FDI information, Right column: with FDI information

Plots of AoA and g-force are give in 5.27 and 5.28 respectively. Again, the AoA plots show that the 15 deg upper limit on AoA was never reached hence the equations of motion in the process model were valid throughout the flight. Also the maximum g-force reached was only 1.5g which is very unlikely to cause structural damage.

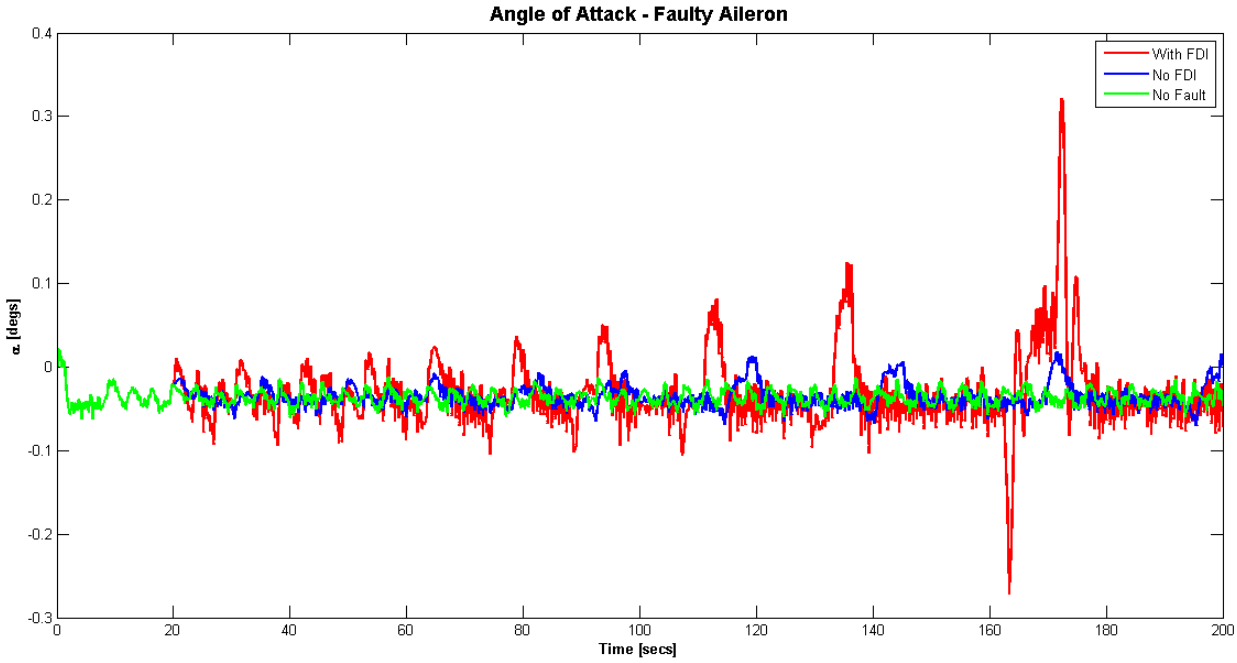

Figure 5.27: Faulty Aileron: Angle of Attack

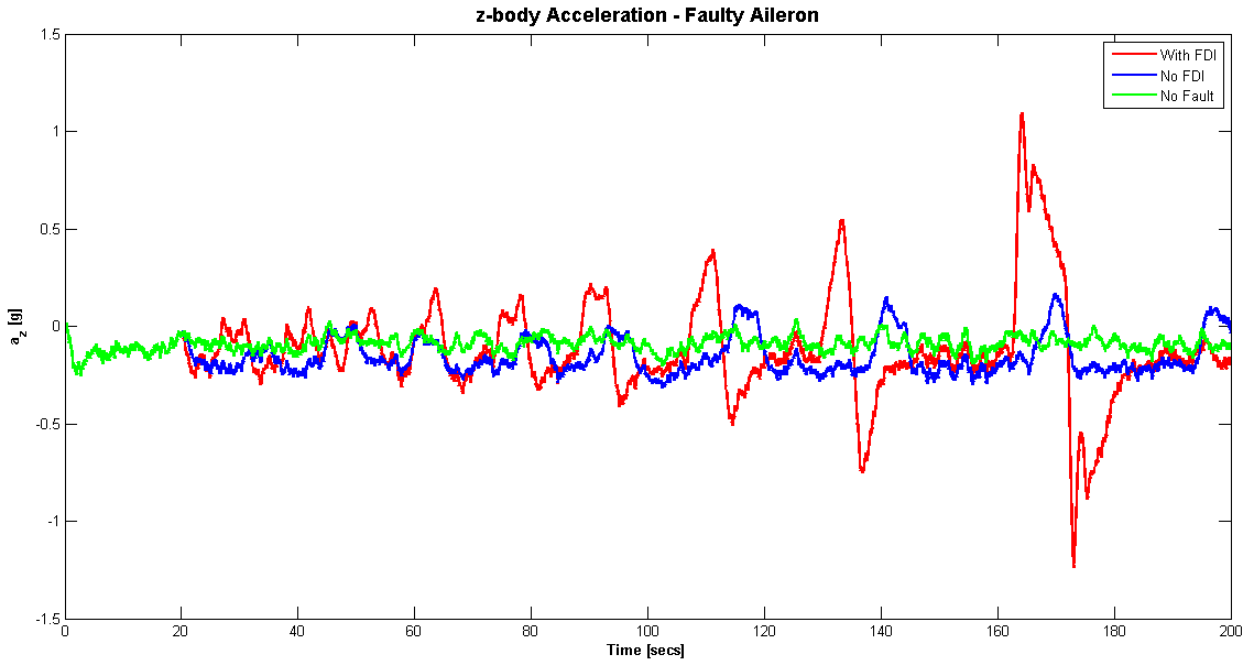

Figure 5.28: Faulty Aileron: g-Force, Body Acceleration a_Z

The trajectory plots of the aircraft are given in figure 5.29. The case where FDI information is provided the aircraft can be seen to deviate slightly off the path. The deviation is not as significant in the event of an aileron fault as the rudder also helps to control the lateral

motion of the aircraft hence providing an extra degree of redundancy. A closer look at the NED states have been presented in figure 5.30. The plots clearly show that upon zooming in on the states there are differences in the no fault case and the with and without FDI cases. Although the trajectory plot shows very good tracking, upon closer investigation there are discrepancies especially in the Down direction.

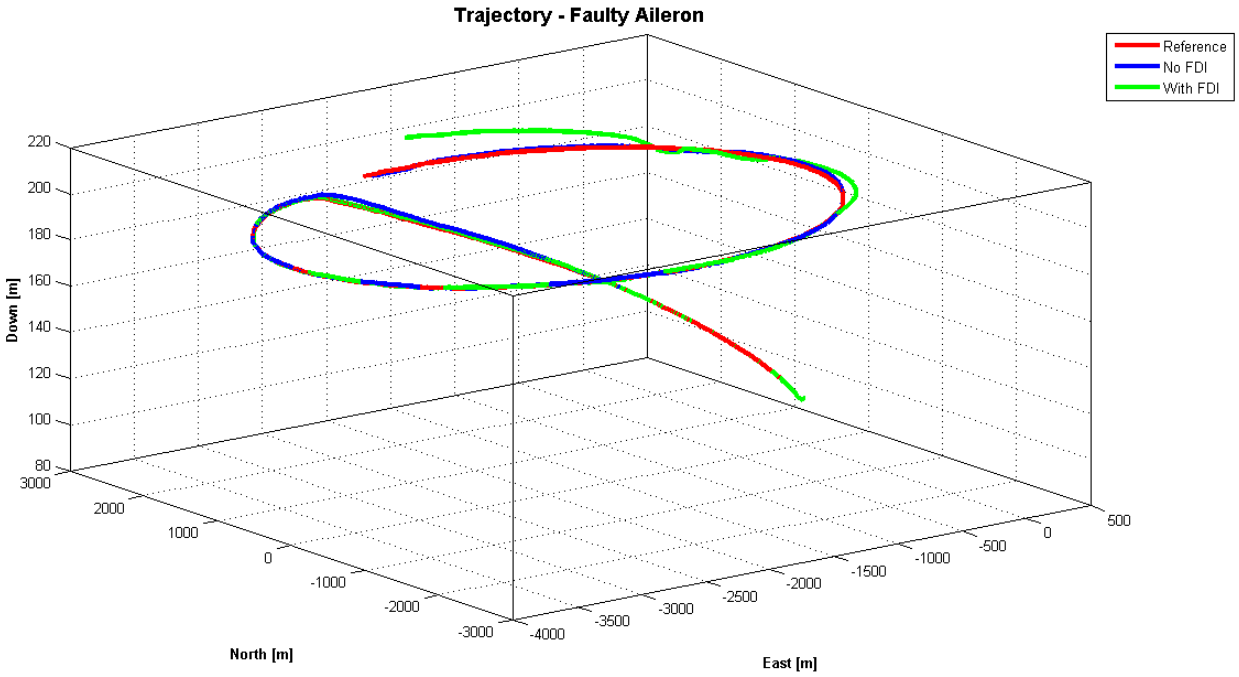

Figure 5.29: Faulty Aileron: 6DoF Trajectory

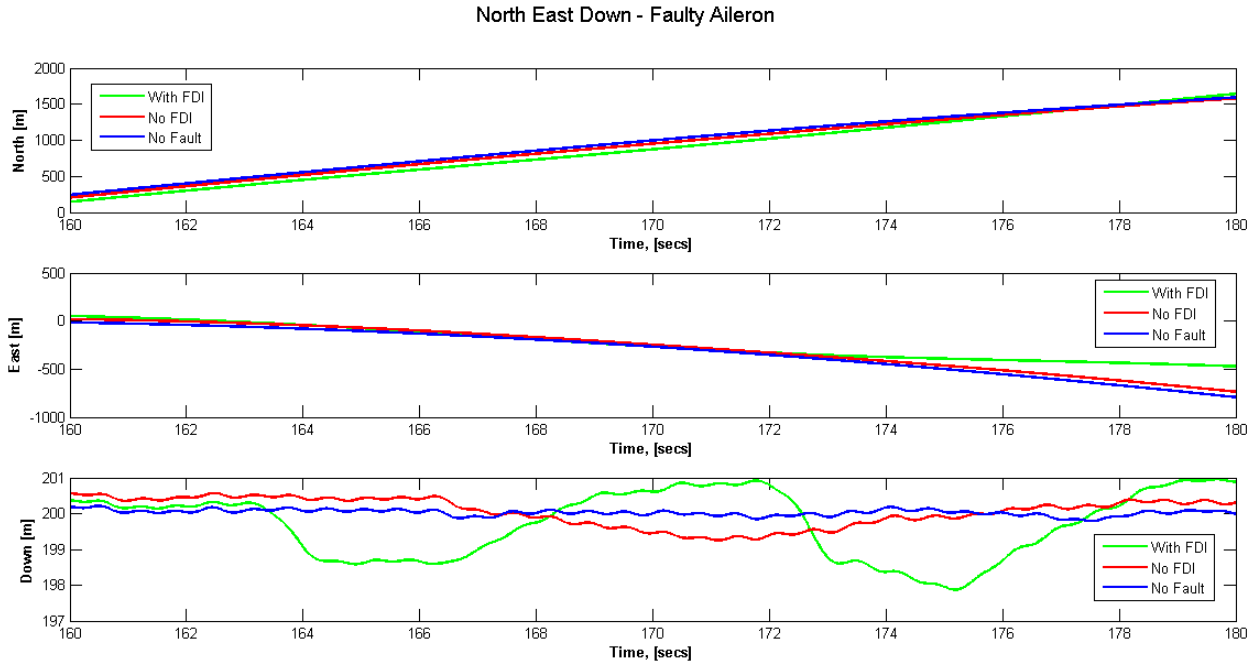

Figure 5.30: Faulty Aileron: NED

5.5.2.3 Scenario 3: Faulty Rudder

With a 60% reduction in efficiency in the rudder the resulting control surface activity is provided in figure 5.31. Again, as the elevator has very little influence on lateral motion, there is very little change in elevator activity with no difference between the no FDI and with FDI cases. The rudder is pushed to its lower limit and the aileron deflection increases in the negative direction causing the aircraft to bank more to the left.

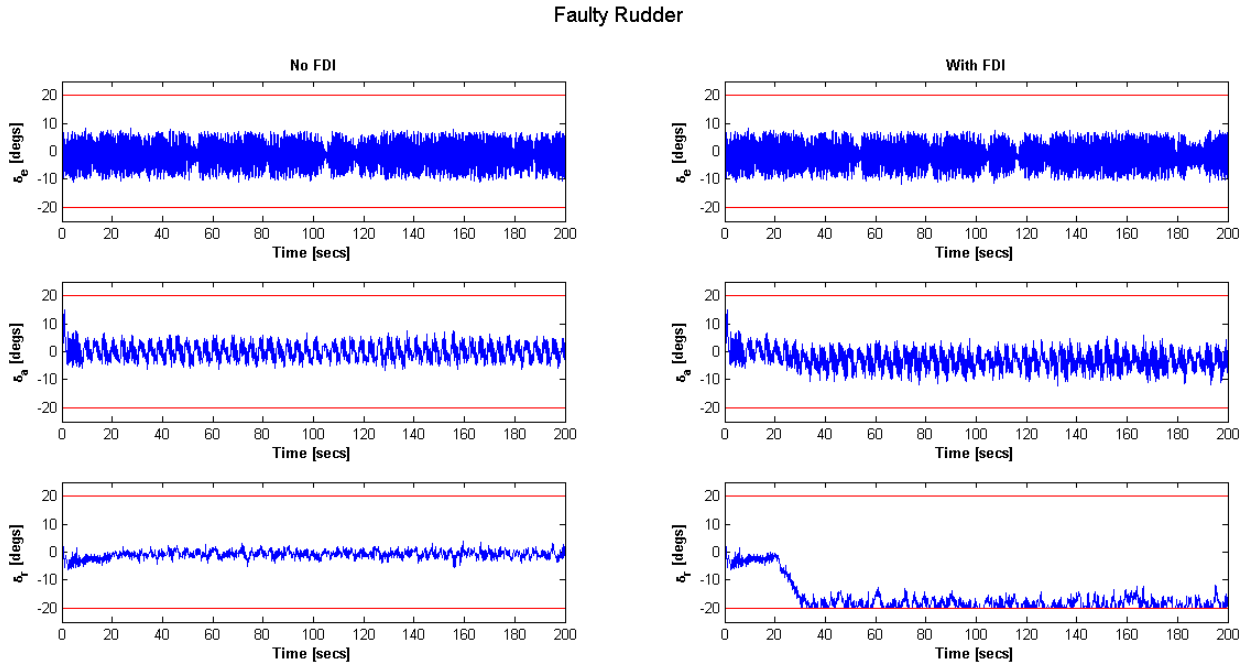

Figure 5.31: Faulty Rudder: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information

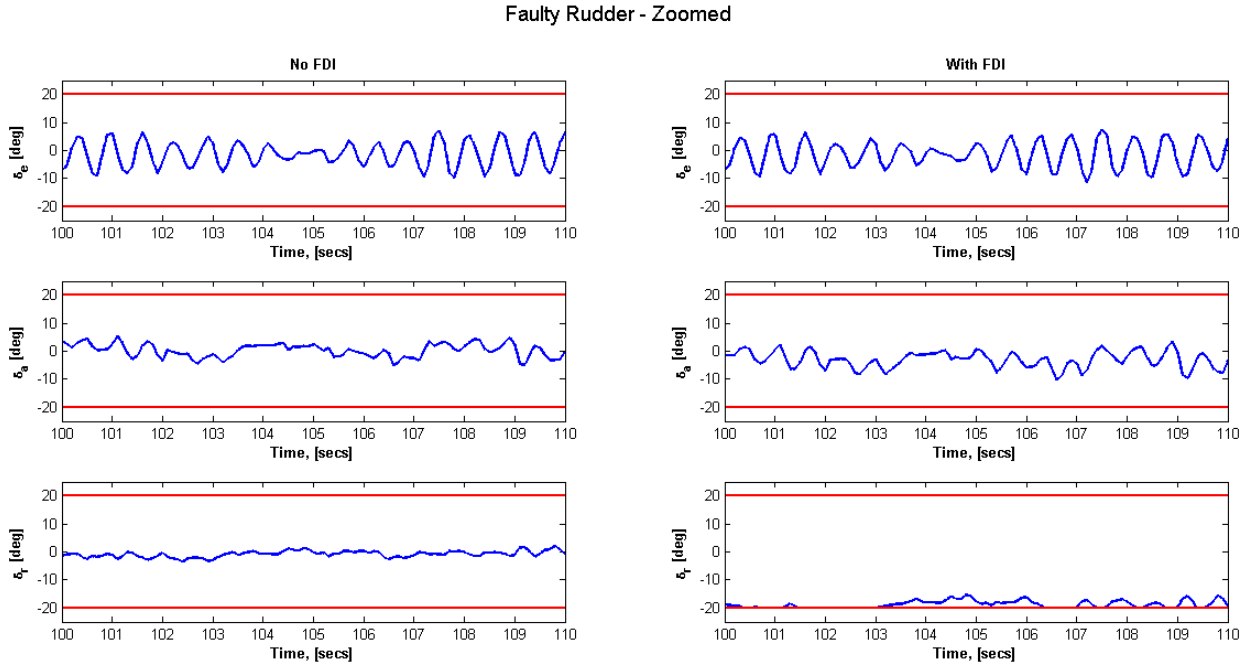

Figure 5.32: Faulty Rudder: Control Surface Activity, constraints (red), control surface activity (blue). Left column: no FDI information, Right column: with FDI information - Zoomed

Figure 5.32 shows the elevator activity between 100 secs and 110 secs, that is, 80 secs after the

fault has occurred. Zooming in at this time the plot shows that the rate constraints on the control surfaces are being respected. Again, this is as to be expected as the actuator dynamics have been modelled in the controller as well as in the plant model.

A faulty rudder had no effect on the angular rate demands (figure 5.33). Tracking performance was the same both with and without FDI information. This is translated in the trajectory plots of figure 5.34 which show that the aircraft closely follows the flight path with and without FDI information. A closer look at the NED states (figure 5.35) shows that tracking performance is excellent in the North and East direction, however there are slight variations present in the Down direction.

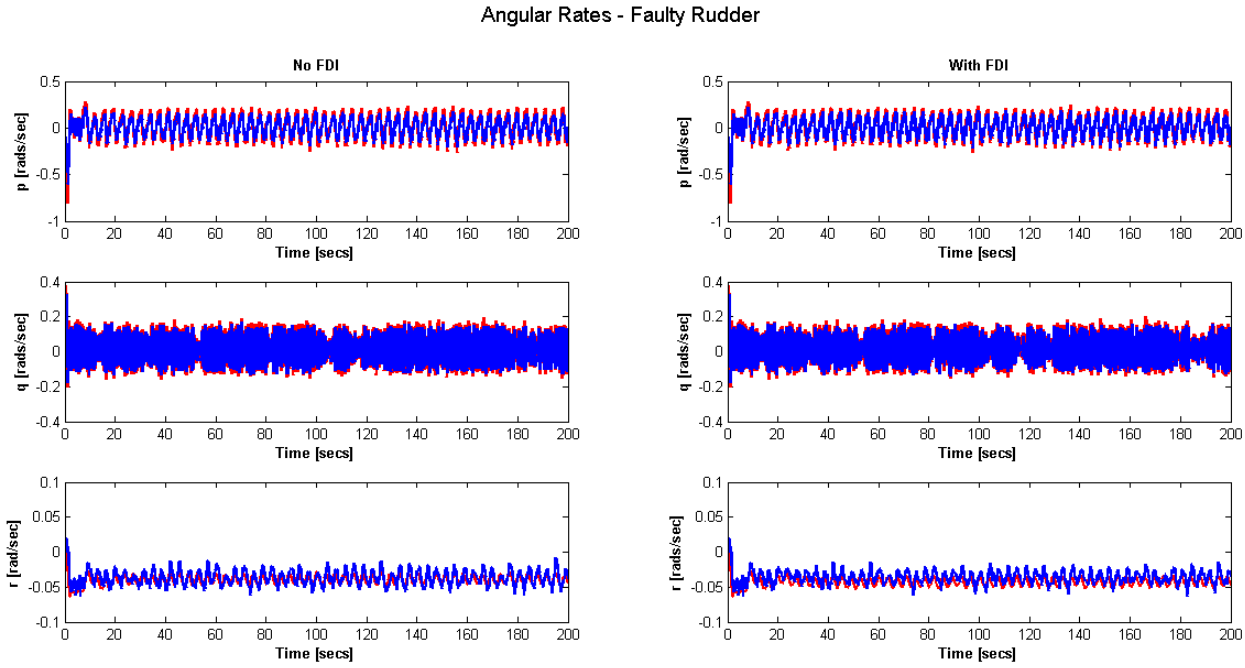

Figure 5.33: Faulty Rudder: Angular Rates, demanded (red), actual (blue). Left column: no FDI information, Right column: with FDI information

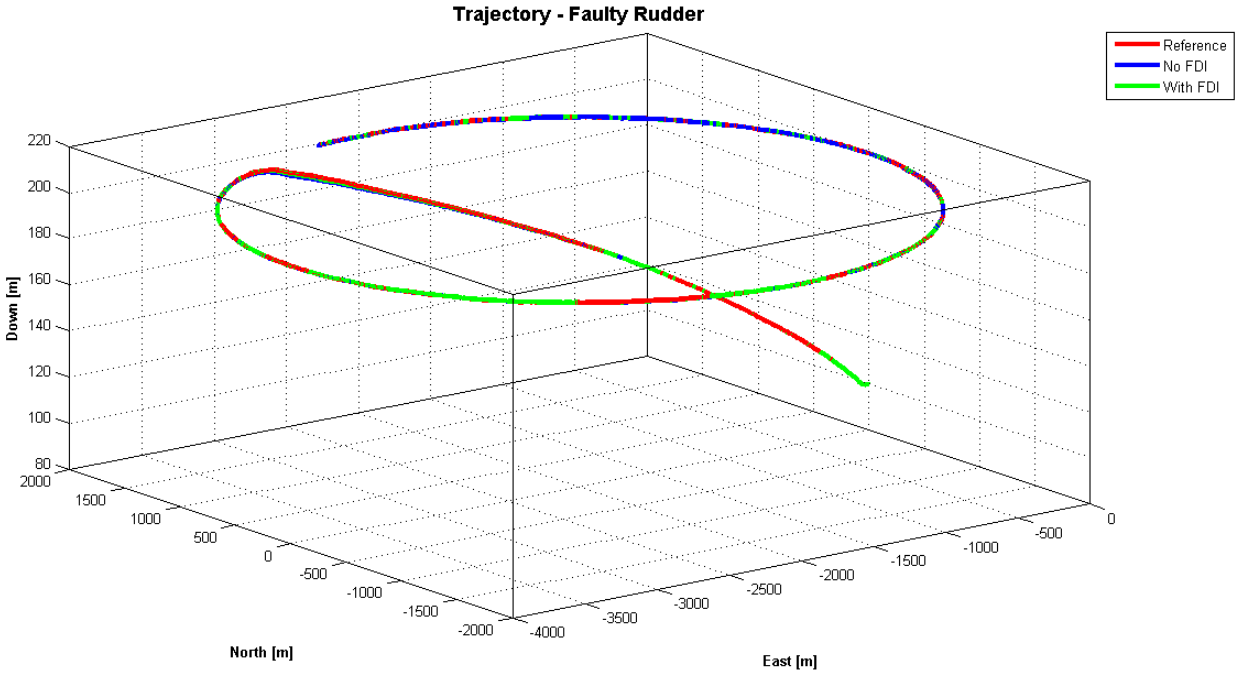

Figure 5.34: Faulty Rudder: 6DoF Trajectory

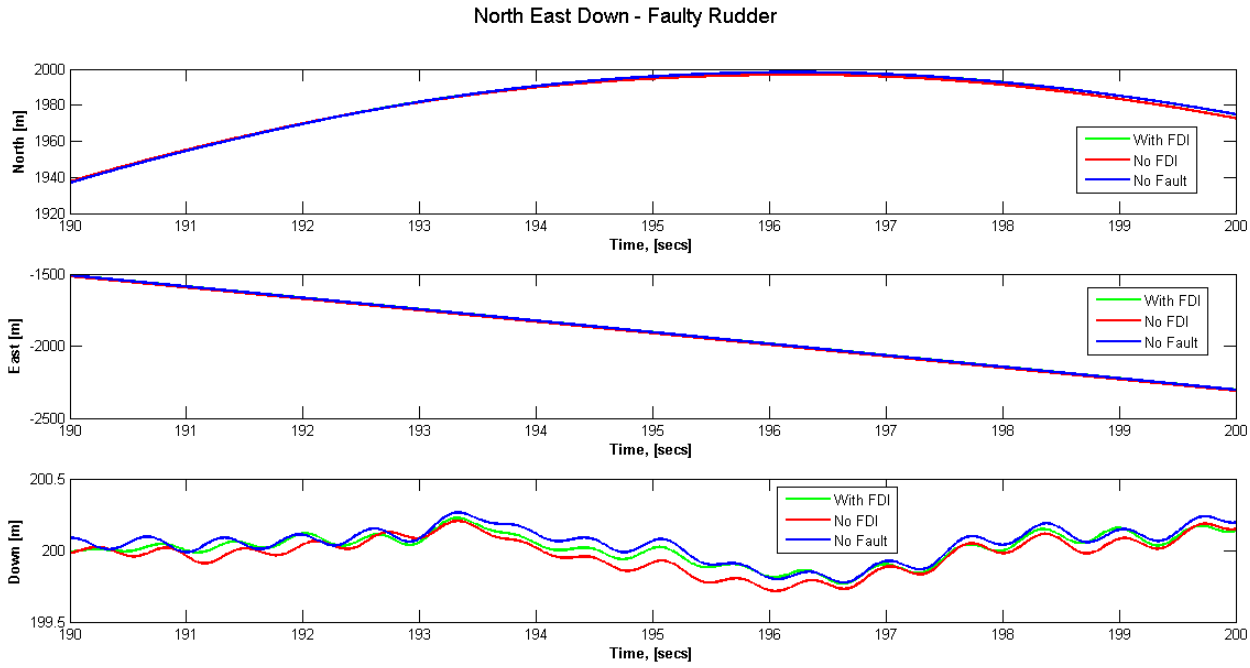

Figure 5.35: Faulty Rudder: NED

Plots of AoA and g-force are give in 5.36 and 5.37 respectively. Again the AoA plots show that the 15 deg upper limit on AoA was never reached, so the equations of motion in the process model were valid throughout the flight. Also the maximum g-force reached was only 1.5g which

is very unlikely to cause structural damage.

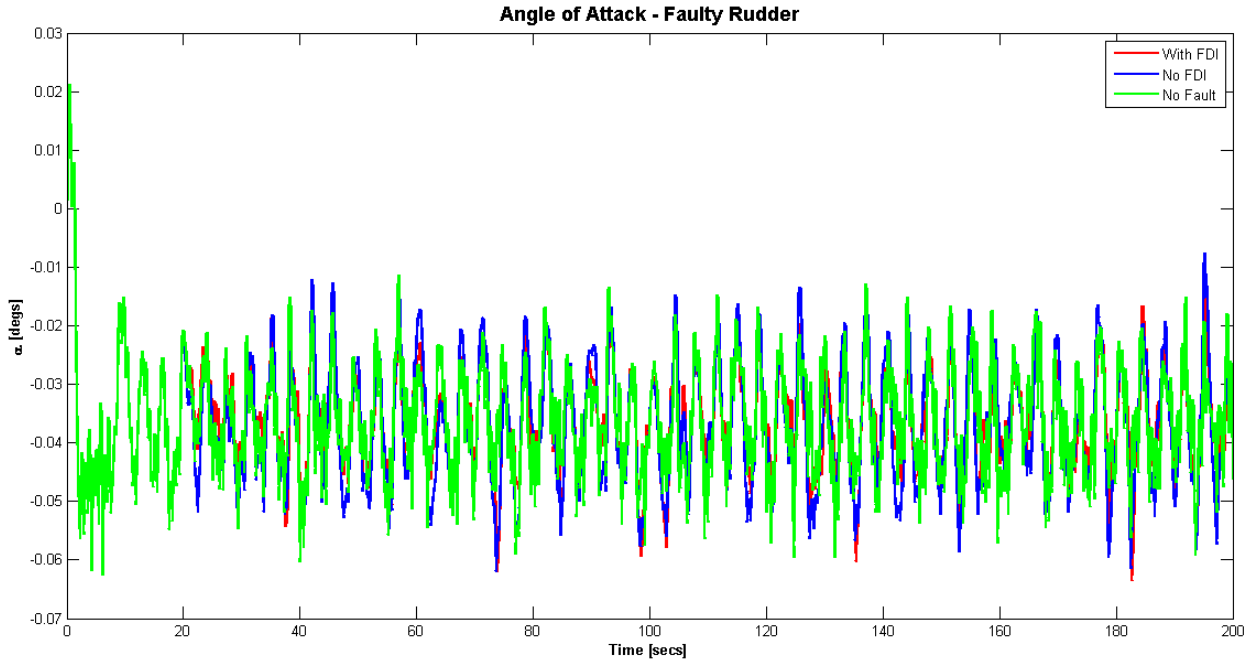

Figure 5.36: Faulty Rudder: Angle of Attack

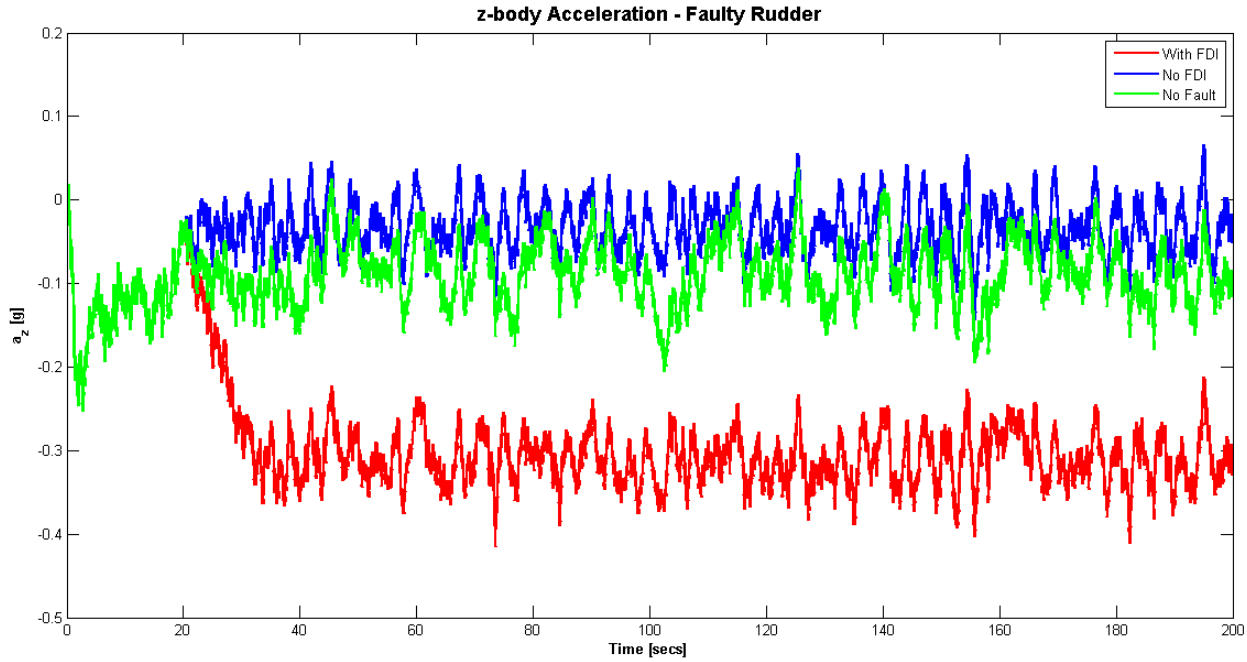

Figure 5.37: Faulty Rudder: g-Force, Body Acceleration a_Z

5.5.2.4 Findings

The results of the 6DoF analysis show that NMPC design as a fault tolerant controller is viable, and that in the absence of FDI information the controller is capable of allocating control authority to the appropriate actuators to fly the aircraft on the given flight path. This illustrates the inherent fault tolerant capabilities of MPC. Turning on FDI updates improved the tracking performance of the controller. The results did show however that unless a quantity is penalised in the cost function, and/or constraints are applied, the controller will push the limits to achieve the desired outcome. In this case the controller was specifically designed to track angular rate demands, so providing FDI information resulted in an increase in tracking performance of angular rates in the event of a control surface fault.

The next section will look at the design of an FDI filter to be incorporated into the FTC developed in this section.

5.6 6DoF Fault Detection and Identification

The fault detection concepts covered in chapter 4 are implemented here for the full 6DoF aircraft model and designs for the UKF are presented. A proportional integral derivative (PID) controller for the aircraft is designed and implemented.

A traditional PID controller was used to control the aircraft through the range of manoeuvres required to test and tune the filter. The PID control method, although not optimal in terms of performance, was quick to implement and tune to the level required.

5.6.1 PID Controller Design

The PID controller assumes a fixed structure where an aileron is the only control surface which can produce a roll manoeuvre, pitch control is performed only by the elevators and yaw movement is provided only by the rudder. The PID controller forms the inner-loops of the system given in figure 5.2. A Simulink diagram of the PID based inner loops is given in figure 5.38.

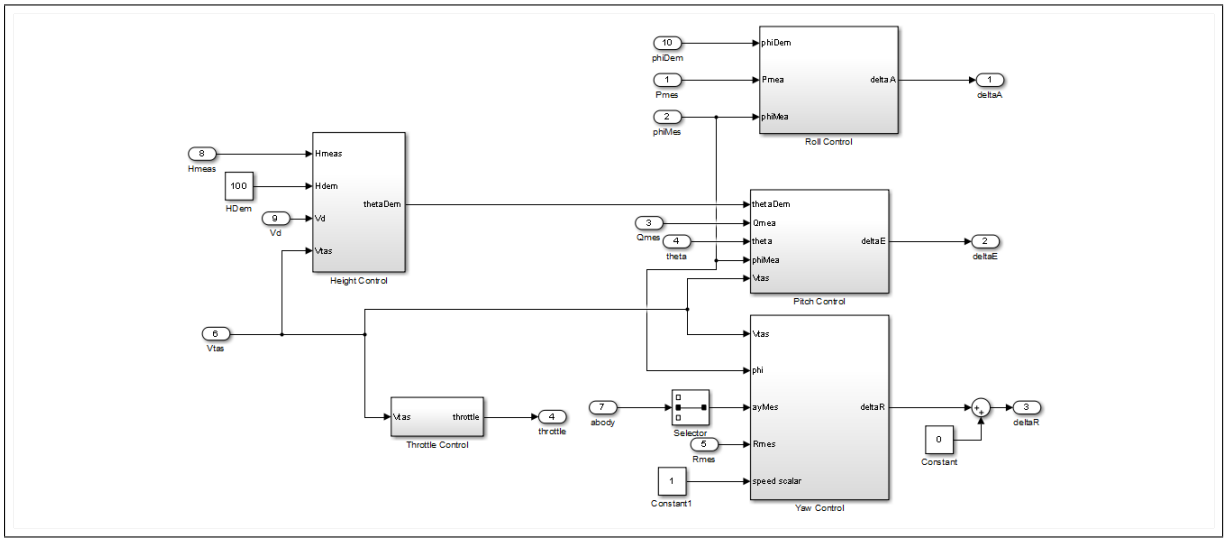

Figure 5.38: Inner Loop with PID Control

The height control loops and the throttle control also form a part of the inner loops.

5.6.1.1 Height Control Loop

The Simulink model of the height control loop is given in figure 5.39. There are four inputs, the vertical speed V_D, the height demand H_{dem}, the measured or actual height of the aircraft H_{meas} and the true airspeed V_{TAS}. The objective of the height control loop is to calculate the pitch angle required to maintain the height demand. The error between the measured and actual height is given by:

$$e_H = H_{\text{dem}} - H_{\text{meas}}. \tag{5.56}$$

The output of the PID controller is the rate of climb $V_{\mathbf{ROC}}$:

$$V_{\mathbf{RoC}} = K_{dP}\, V_D + K_P\, e_H + K_I \int e_H(t)\, \mathrm{d}t, \tag{5.57}$$

where:

$$\begin{aligned} K_{dP} &= 0.25, \quad \text{Proportional gain for vertical speed,} \\ K_P &= 0.5, \quad \text{Proportional error gain,} \\ K_I &= 0.025, \quad \text{Integral error gain.} \end{aligned}$$

The climb rate to pitch angle conversion is given by:

$$\theta = \frac{V_{\text{RoC}}}{V_{\text{TAS}}}. \tag{5.58}$$

The pitch angle demand is constrained between 0 and 0.25 rads hence θ_{dem} is given by:

$$\theta_{\text{dem}} = \begin{cases} \theta, & \text{if } 0\,\text{rad} \leq \theta \leq 0.25\,\text{rads}, \\ 0\,\text{rad}, & \text{if } \theta < 0\,\text{rads}, \\ 0.25\,\text{rads}, & \text{if } \theta > 0.25\,\text{rads}. \end{cases} \tag{5.59}$$

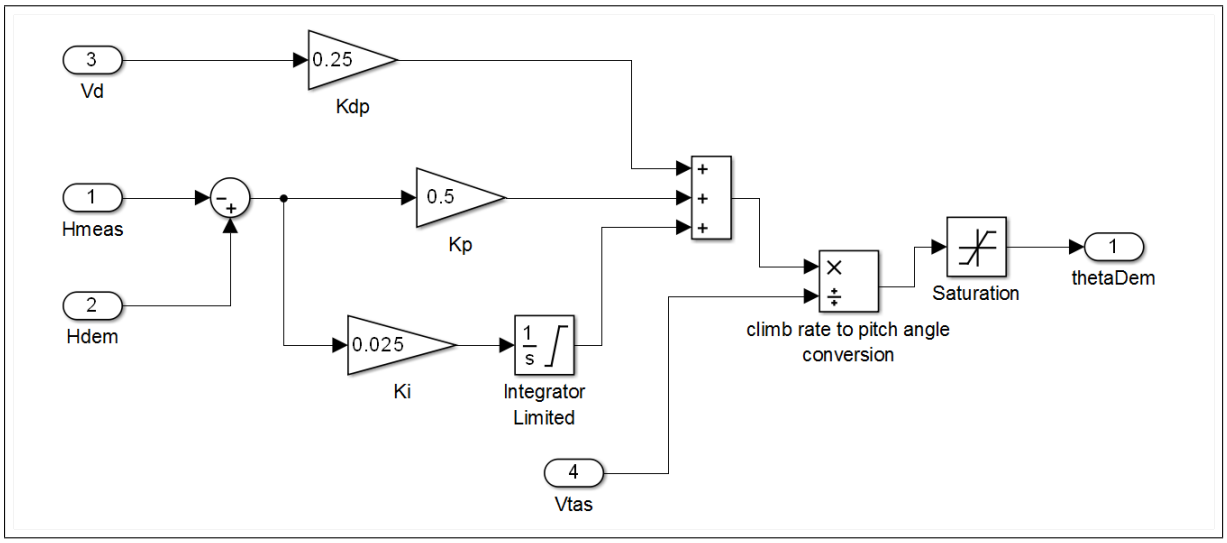

Figure 5.39: Height Control Loop

5.6.1.2 Throttle Control Loop

The throttle control loop is given in figure 5.40, the objective of which is to calculate the throttle needed to maintain the true airspeed demand. The error in true airspeed is given by:

$$e_{V_{\mathbf{TAS}}} = (V_{\text{TAS}})_{\text{dem}} - V_{\text{TAS}}. \tag{5.60}$$

The output of the PID controller is u_{th}:

$$u_{th} = K_P\, e_{V_{\mathbf{TAS}}} + K_I \int e_{V_{\mathbf{TAS}}}(t)\,\text{dt} + K_D \frac{d(e_{V_{\mathbf{TAS}}})}{dt} + \delta_{\text{th trim}}, \tag{5.61}$$

where:

$$\begin{aligned} K_P &= 0.1, && \text{Proportional error gain,} \\ K_I &= 0.01, && \text{Integral error gain,} \\ K_D &= 0.05, && \text{Derivative error gain,} \\ \delta_{\text{th trim}} &= 0.5, && \text{Throttle trim.} \end{aligned}$$

Speed is controlled by applying anywhere between 0% throttle to 100% throttle hence:

$$\delta_{\text{th}} = \begin{cases} u_{\text{th}}, & \text{if } 0 \leq u_{\text{th}} \leq 1, \\ 0, & \text{if } u_{\text{th}} < 0, \\ 1, & \text{if } u_{\text{th}} > 1. \end{cases} \tag{5.62}$$

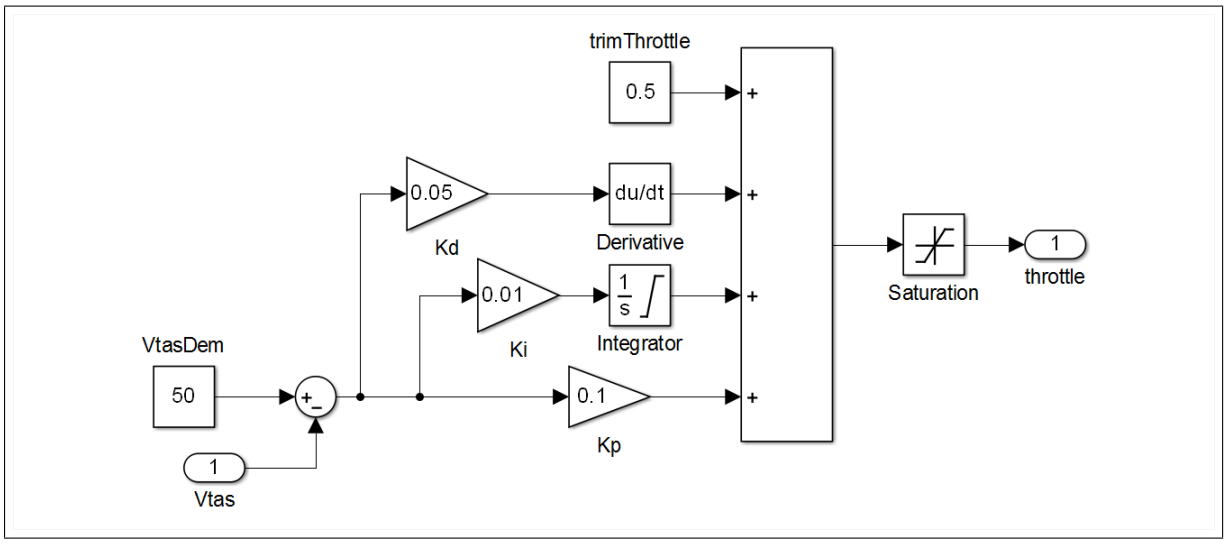

Figure 5.40: Throttle Control Loop

5.6.1.3 Roll Control

The roll control loop is given in figure 5.41. The inputs are roll angle demand ϕ_{dem}, measured roll angle ϕ_{meas} and measured roll rate p_{meas}. The overall aim of this loop is to calculate the aileron deflection needed to maintain a roll angle demand. The error between the roll angle demand and measured roll angle is given by:

$$e_\phi = \phi_{\text{dem}} - \phi_{\text{meas}}. \tag{5.63}$$

The demanded roll rate is calculated via:

$$P_{\text{dem}} = K_{\phi}\, e_{\phi}, \tag{5.64}$$

where K_{ϕ} is a gain factor used to convert the roll angle error to a roll rate and is equal to 2.

The error in roll rate is:

$$e_p = p_{\text{dem}} - p_{\text{meas}}. \tag{5.65}$$

The output of the controller is aileron deflection given by:

$$\delta_a = K_{\text{FF}}\, p_{dem} + K_P\, e_p, \tag{5.66}$$

where:

$$K_{\text{FF}} = 0.8, \quad \text{Feedforward gain,}$$
$$K_P = 2.2, \quad \text{Proportional error gain.}$$

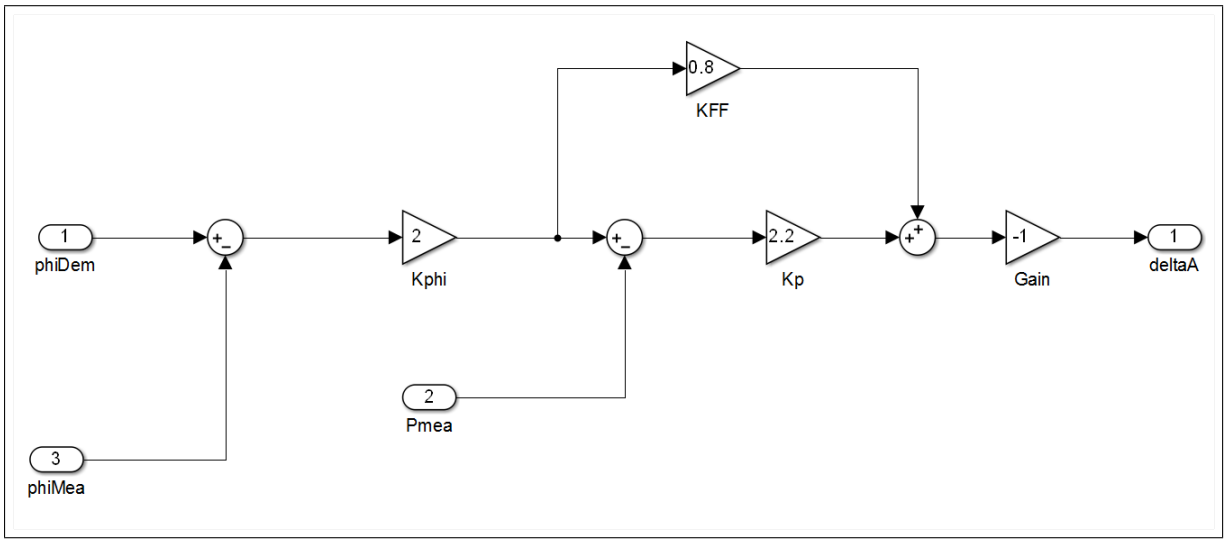

Figure 5.41: Roll Control Loop

5.6.1.4 Pitch Control

The pitch control loop is given in figure 5.42. The inputs are measured roll angle ϕ_{meas}, true airspeed V_{TAS}, pitch angle demand θ_{dem}, measured pitch angle θ_{meas} and measured pitch rate q_{meas}. The overall objective is to calculate the correct elevator deflection required to maintain the demanded pitch and pitch rate. The pitch angle error is given by:

$$e_\theta = \theta_{\text{dem}} - \theta_{\text{meas}}. \tag{5.67}$$

The demanded pitch rate is given by:

$$q_{\text{dem}} = K_\theta \, e_\theta + |f\left(\phi, V_{\text{TAS}}\right)|, \tag{5.68}$$

where:

$$f\left(\phi_{\text{meas}}, V_{\text{TAS}}\right) = \dot{\phi}_{\text{dem}} = \frac{g \tan\left(\phi_{\text{meas}}\right) \sin\left(\phi_{\text{meas}}\right)}{V_{\text{TAS}}}. \tag{5.69}$$

Here g is the acceleration due to gravity, $9.81\,\text{m/s}^2$. The function given in equation (5.69) calculates the additional pitch rate required for a constant height coordinated turn.

The error in pitch rate is given by:

$$e_q = q_{\text{dem}} - q_{\text{meas}}. \tag{5.70}$$

Finally the elevator deflection is given by:

$$\delta_e = K_{\text{FF}} \, q_{\text{dem}} + K_p \, e_q + K_P \, K_I \int e_q(t) \, \text{dt}, \tag{5.71}$$

where:

$K_{\text{FF}} = 0.01,$	Feedforward gain,
$K_\theta \ = 2,$	Gain required to convert to pitch rate,
$K_P \ = 1.2,$	Proportional error gain.

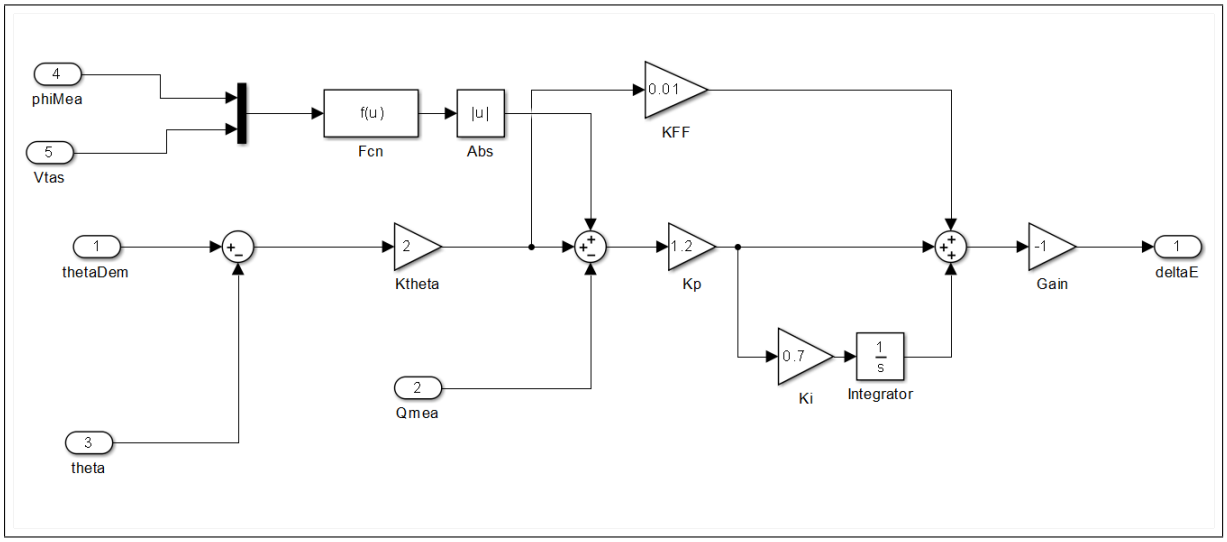

Figure 5.42: Pitch Control Loop

5.6.1.5 Yaw Control

The yaw control loop is given in figure 5.43. The inputs are true airspeed V_{TAS}, measured roll angle ϕ_{meas}, measured yaw rate r_{meas}, body acceleration a_y and a speed scalar constant. The overall objective of the yaw control loop is to maintain zero sideslip. The yaw rate error due to a turn is given by:

$$e_{r(turn)} = (r_{\text{meas}} - r_{\text{dem}}) * h(t), \tag{5.72}$$

where $*$ indicates a convolution operation between the two functions, and the demanded yaw rate is given by equation (5.73):

$$r_{\text{dem}} = K_\phi f_{\dot{\psi}}(\phi, V_{\text{TAS}}), \tag{5.73}$$

$f_{\dot{\psi}}(\phi, V_{\text{TAS}})$ is the yaw rate required to turn and is given by

$$f_{\dot{\psi}}(\phi, V_{\text{TAS}}) = \frac{g \tan\phi \cos\phi}{V_{\text{TAS}}}, \tag{5.74}$$

and $h(t)$ is a high pass filter whose Laplace transform is given by:

$$H(s) = \frac{5s}{5s+1}. \tag{5.75}$$

The highpass filter is there to prevent measurement errors in airspeed and bank angle (roll) producing a steady state sideslip condition.

The yaw rate error is then given by:

$$e_r = -e_{r(turn)} - K_\beta \, a_y, \tag{5.76}$$

where K_β is the gain due to sideslip and is equal to 0.1. Finally the rudder deflection is given by:

$$\delta_r = K_{\text{damp}} \left(K_I \, e_r - e_{r(turn)} \right) \, S, \tag{5.77}$$

$$\begin{aligned} S \quad &= 1, \quad \text{Speed scalar,} \\ K_I \quad &= 1, \quad \text{Integral error gain,} \\ K_{\text{damp}} &= 3, \quad \text{Yaw damper.} \end{aligned}$$

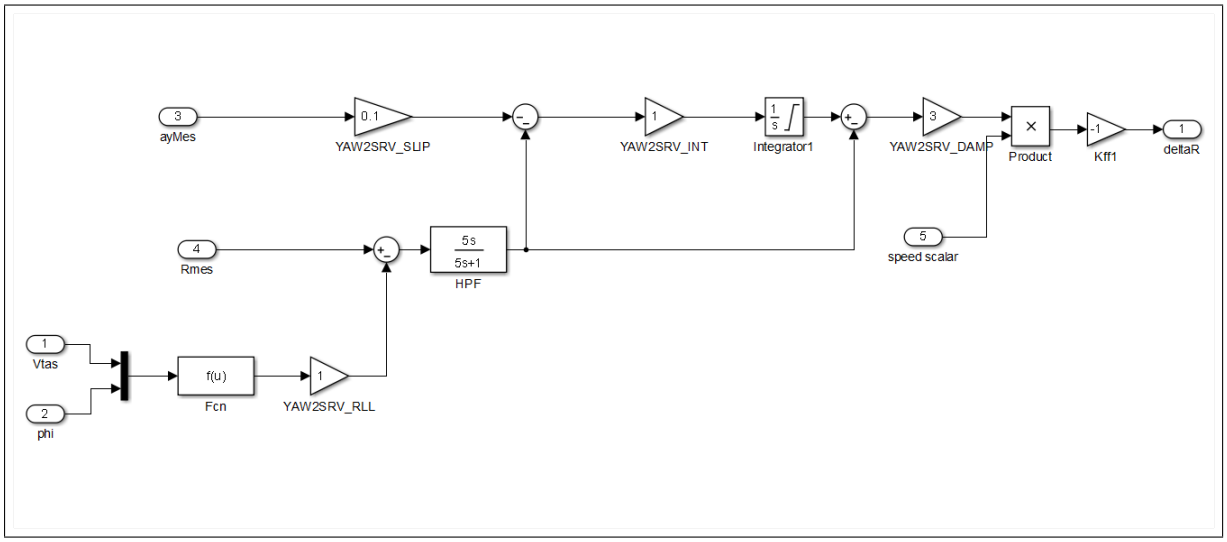

Figure 5.43: Yaw Control Loop

5.6.2 Filter Design

The proposed fault detection scheme is based on the principle that a failure in any one of the control surfaces would directly affect the corresponding control derivative. Hence changes in the control derivatives would indicate a fault has occurred, while at the same time the filter would provide the controller with estimates of the derivatives. Furthermore, up to date estimates of the derivatives will allow the MPC controller to perform at its optimum.

The force and moment equations given in section 5.3.1.2 show that there are a total of 24 control derivative. These are listed in Table 5.6.

Table 5.6: Control Derivatives

Derivative	**Value**	**Derivative**	**Value**
CX_{dE1}	9.5×10^{-4}	Cl_{dR2}	4.5×10^{-6}
CX_{dE2}	8.5×10^{-7}	Cl_{dE1}	5.24×10^{-5}
CY_{dE1}	1.75×10^{-4}	Cm_{dE1}	6.54×10^{-3}
CY_{dR1}	1.55×10^{-3}	Cm_{dE2}	8.49×10^{-5}
CY_{dR2}	8×10^{-6}	Cm_{dE3}	3.74×10^{-6}
CZ_{dE1}	4.76×10^{-3}	Cm_{dA1}	3.5×10^{-5}
CZ_{dE2}	3.3×10^{-5}	Cn_{dA1}	1.4×10^{-5}
CZ_{dA1}	7.5×10^{-5}	Cn_{dA2}	7.0×10^{-6}
Cl_{dA1}	6.1×10^{-4}	Cn_{dE1}	8.73×10^{-5}
Cl_{dA2}	2.5×10^{-5}	Cn_{dE2}	8.7×10^{-6}
Cl_{dA3}	2.6×10^{-6}	Cn_{dR1}	9.0×10^{-4}
Cl_{dR1}	-2.3×10^{-4}	Cn_{dR2}	4.0×10^{-6}

To test the filters the aircraft was required to achieve the roll angle demands given in figure 5.44.

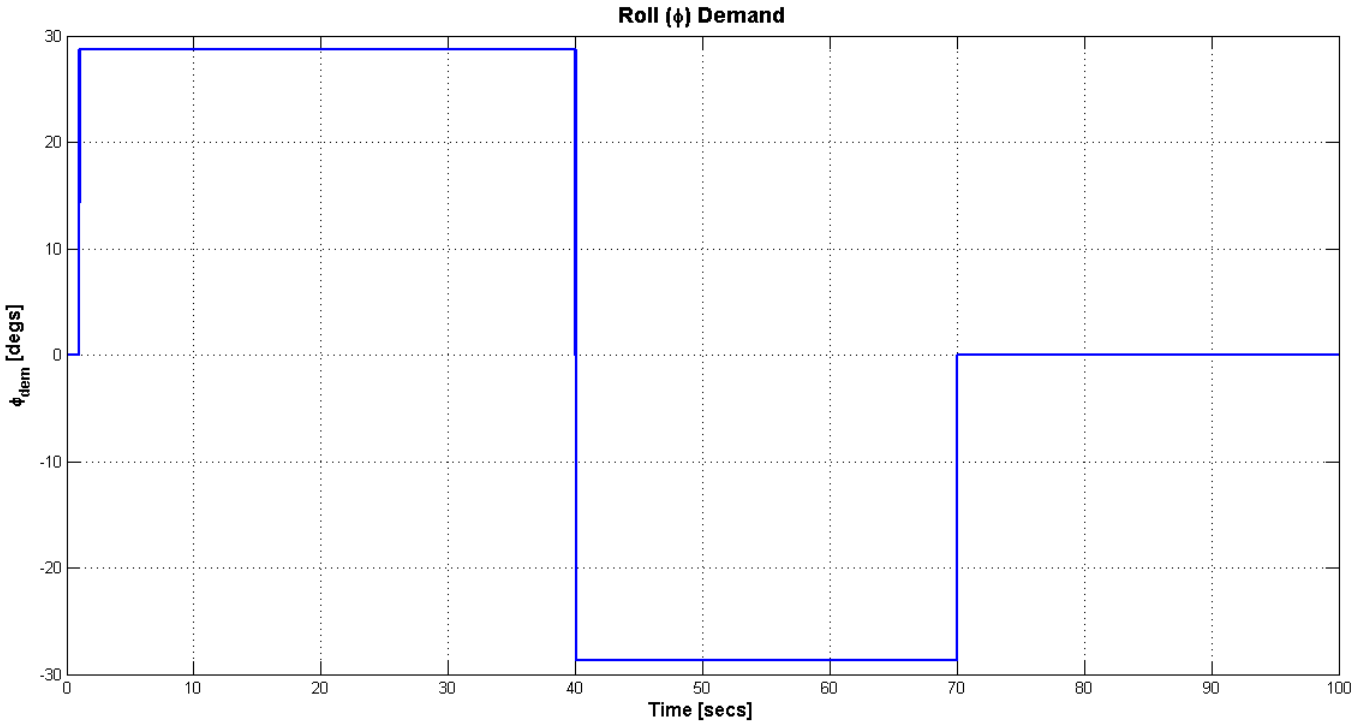

Figure 5.44: 6DoF Motion Filter Tests - Roll Angle Demands

Initially a 30 state UKF filter was designed where the states comprised of three accelerations (a_x, a_y, a_z), three angular rates (p, q, r) and the 24 control derivatives given above. The measurements were of the body acceleration and angular rates (as would be provided by an IMU sensor). All the derivatives were normalised to 1 hence the states of the control derivative were set to 1. The results of the acceleration and angular rate innovations are given in figures 5.45 and 5.46 respectively. The results show that the filter does an excellent job of predicting the accelerations and angular rates as the innovations filter predictions align perfectly with the measurements of acceleration and angular rates. The estimates of the control derivatives are shown in figure 5.47. Since all derivatives were normalised the estimates should each have a value of 1. However, as the plot shows, the filter is unable to correctly estimate the value of all the derivatives, as many of the states in the filter are unobservable.

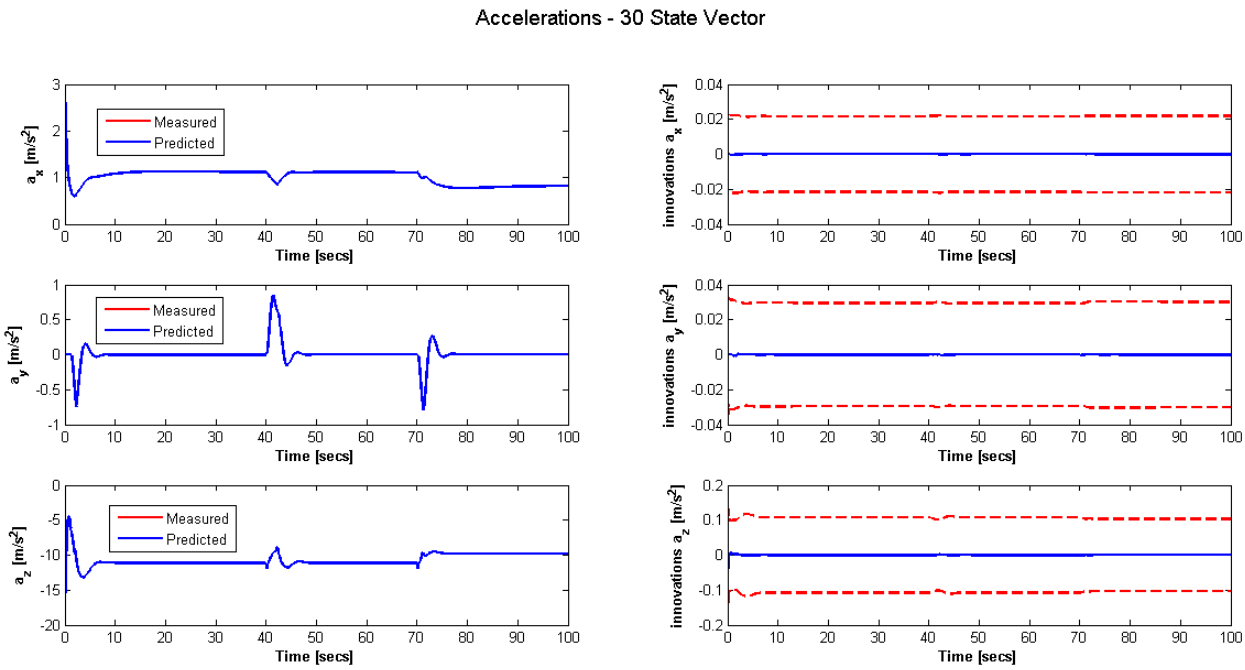

Figure 5.45: Accelerations - 30 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

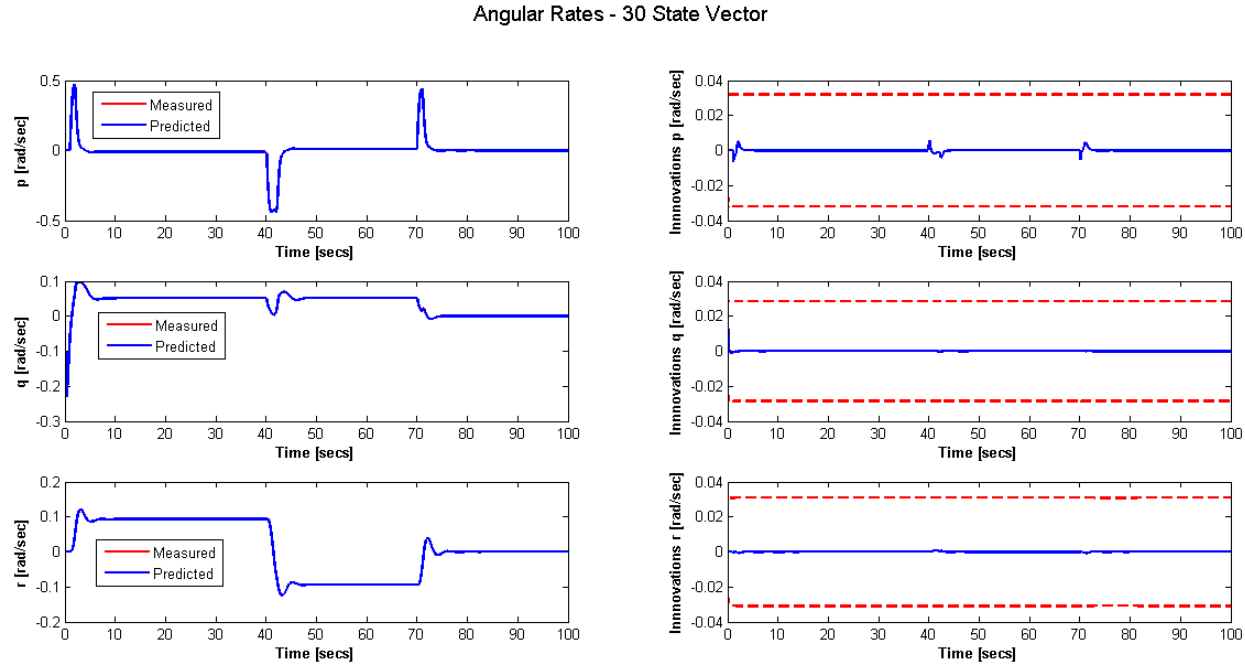

Figure 5.46: Angular Rates - 30 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

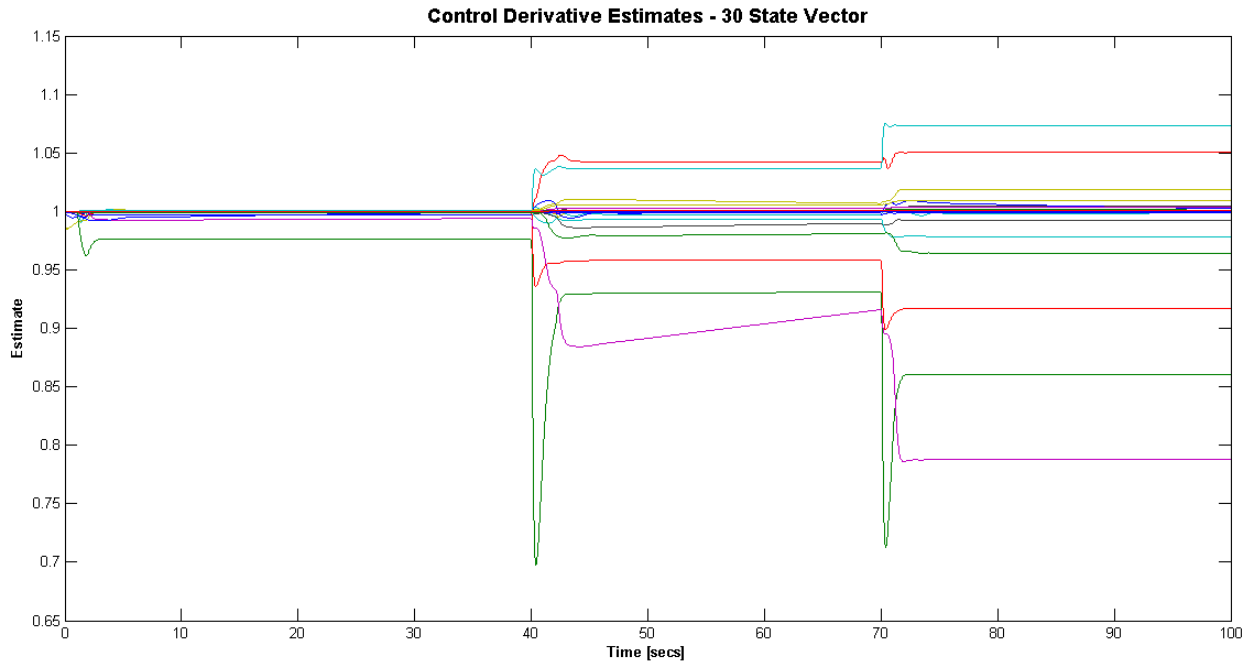

Figure 5.47: Control Derivative Estimates - 30 State Vector. Each line corresponds to a normalised value of a control derivative estimate and should have a value of 1.

To address the issue of observability the number of states was reduced to 19; 3 accelerations, 3 angular rates and 13 control derivatives. This was achieved by having one control derivative estimate per force/moment for a particular control surface. For example, the control derivatives given in table 5.6 show that there are 2 CX control derivatives which are due to the elevator CX_{dE1} and CX_{dE2}; the 19 state filter has only one derivative CX_{dE} used to represent both of the CX derivatives due to the elevator. Thus the contributions of each control surface in the force and moment equations are grouped together in this manner reducing the number of control derivative states from 24 to 13. Figures 5.48 and 5.49 show the acceleration and angular rate estimates respectively produced by the 19 state filter. The results again show close to perfect compliance between prediction and measurement. The control derivative estimates plot given in figure 5.50 show an improvement in estimates (again they should all be equal to 1), however the issue of unobservable states is still evident.

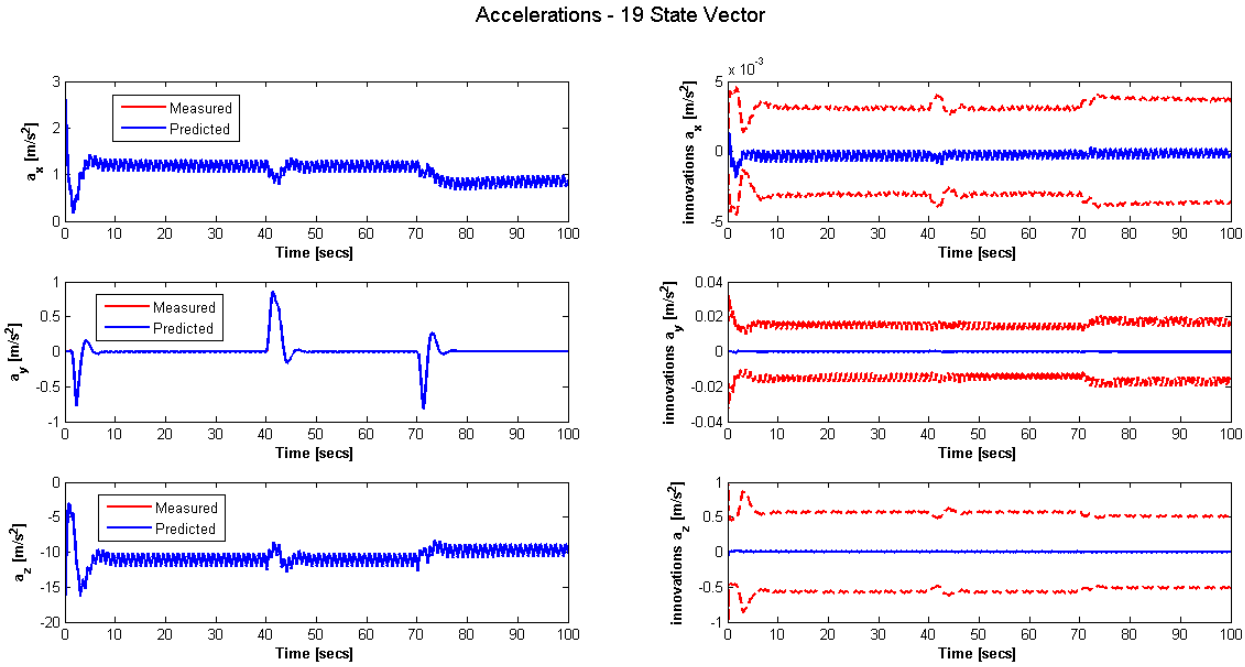

Figure 5.48: Accelerations - 19 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

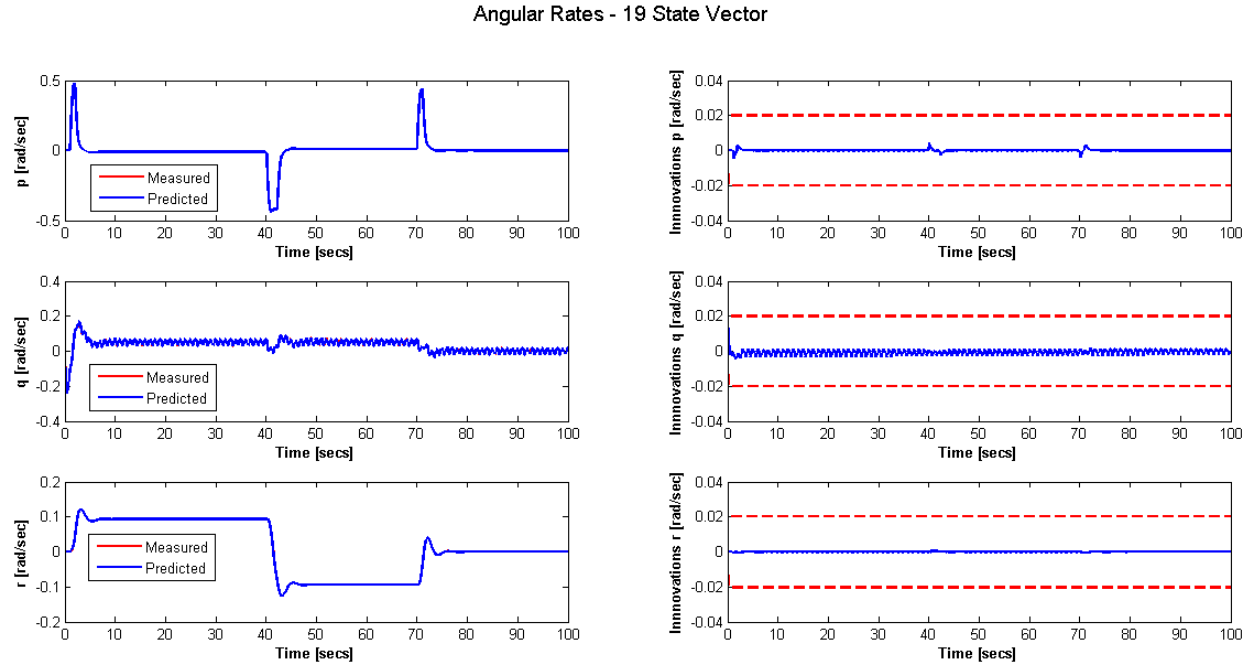

Figure 5.49: Angular Rates - 19 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

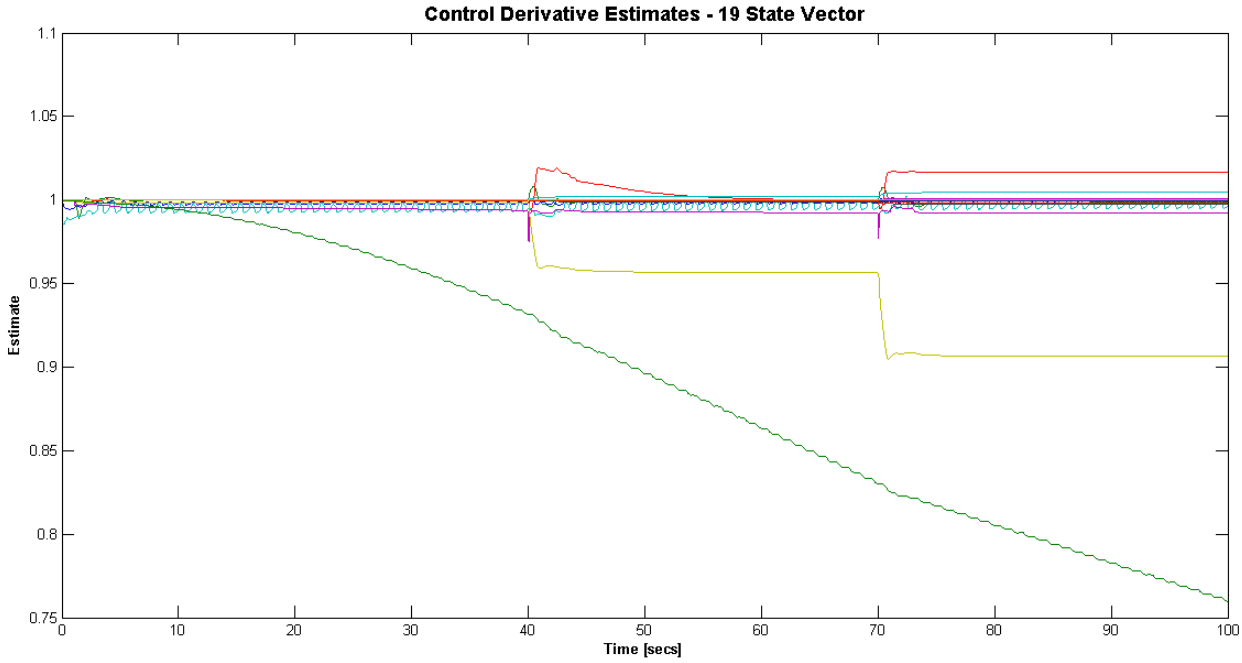

Figure 5.50: Control Derivative Estimates - 19 State Vector. Each line corresponds to a normalised value of a control derivative estimate and should have a value of 1.

To further address this issue the number of filter states was again reduced, from 19 to 12. For control surface failure the most important derivatives are deemed to be Cl_{dA1} for aileron, Cm_{dE1} for elevator and Cn_{dR1} for the rudder and the trim values Cl_0, Cm_0 and Cn_0. From the equations given in section 5.3.1.2 the trim values corresponding to aileron, elevator and rudder are, $Cl_0 = 0$, $Cm_0 = -6.61 \times 10^{-3}$ and $Cn_0 = 0$ respectively.

The plots of acceleration and angular rate estimates for the 12 state filter are given in figures 5.51 and 5.52 respectively. The results of angular rate estimation are excellent however the filter is unable to make correct estimates of acceleration which is to be expected because the other terms are unrelated to force, being all moment related terms. The control derivatives Cl_{dA1}, Cm_{dE1} and Cn_{dR1} are normalised to 1 as is the trim value for Cm_0 and the estimates of these are given in figure 5.53. Results show big discrepancies between the actual and estimated values for the elevator terms.

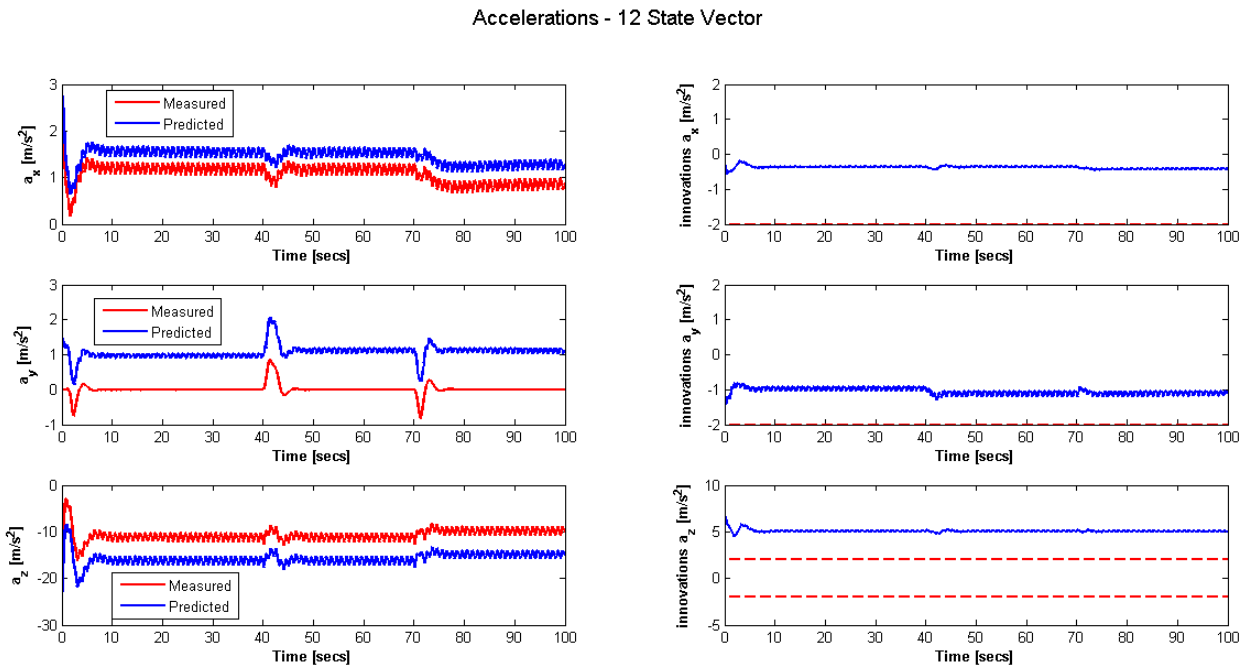

Figure 5.51: Accelerations - 12 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

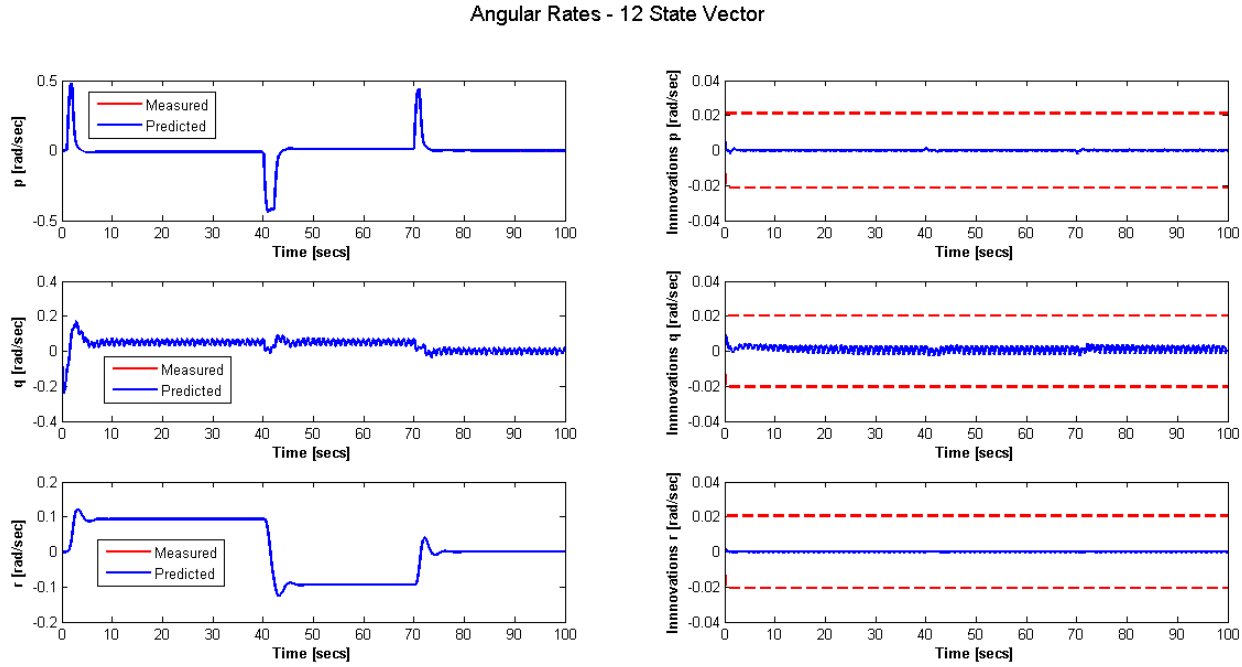

Figure 5.52: Angular Rates - 12 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

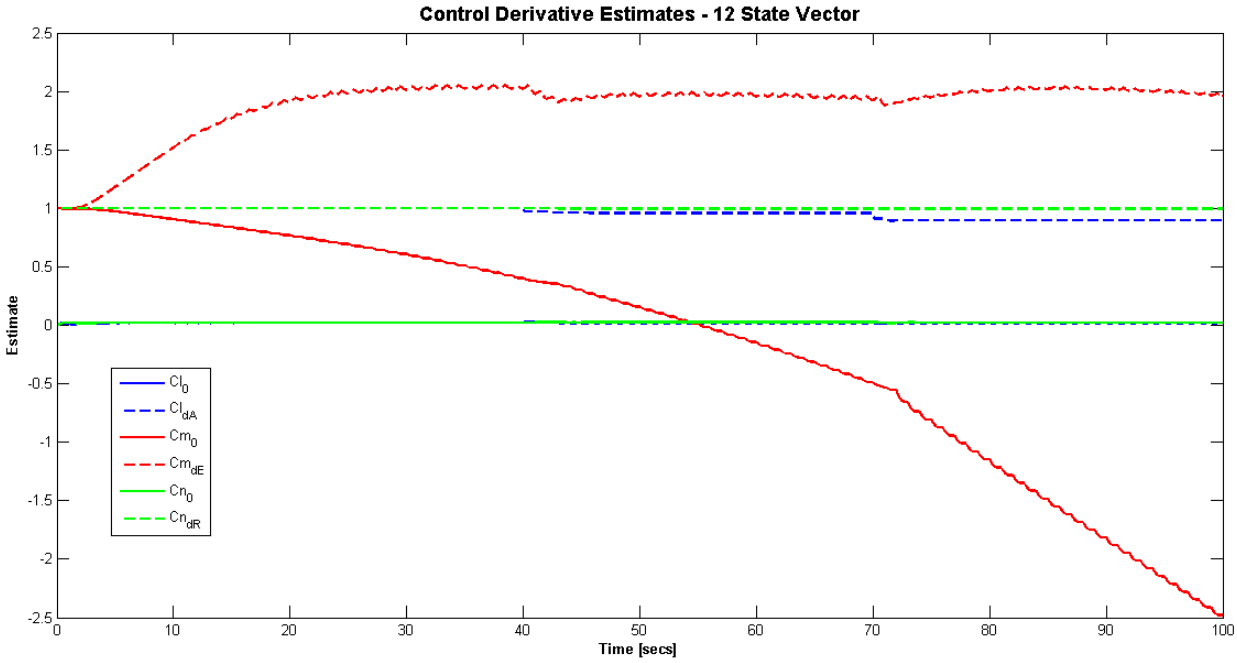

Figure 5.53: Control Derivative Estimates - 12 State Vector. Each line corresponds to a control derivative estimate. Values for Cl_{dA}, Cm_{dE}, Cm_0 and Cn_{dR} have been normalised to have a value of 1, Cl_0 and Cn_0 should both be zero.

Due to the presence of unobservable states the state vector was once more reduced by removing the acceleration terms resulting in a 9 state filter. The measurements supplied to the filter were only of the angular rates. The acceleration terms were removed due to the errors present in the estimates. The angular rate estimates and innovations are presented in figure 5.54 and as expected, show that the filter predictions closely match the measurements. It is evident however from the control derivative and trim estimate plot (figure 5.55) that the observability issue is still present.

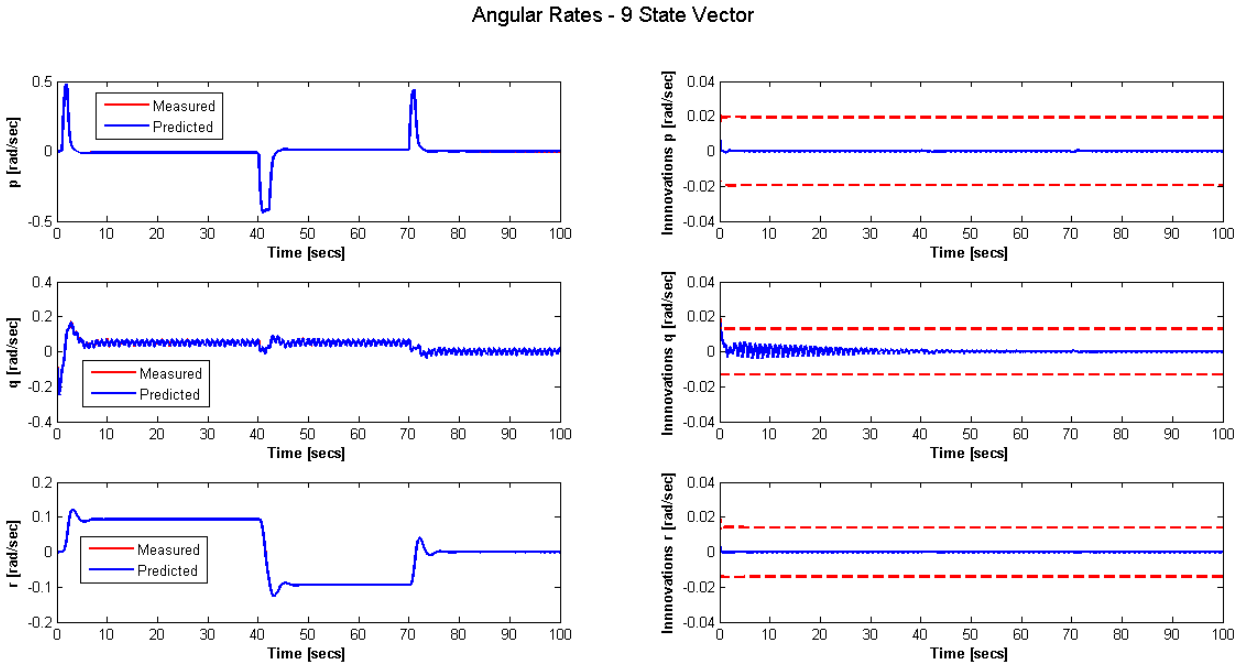

Figure 5.54: Angular Rates - 9 State Vector, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

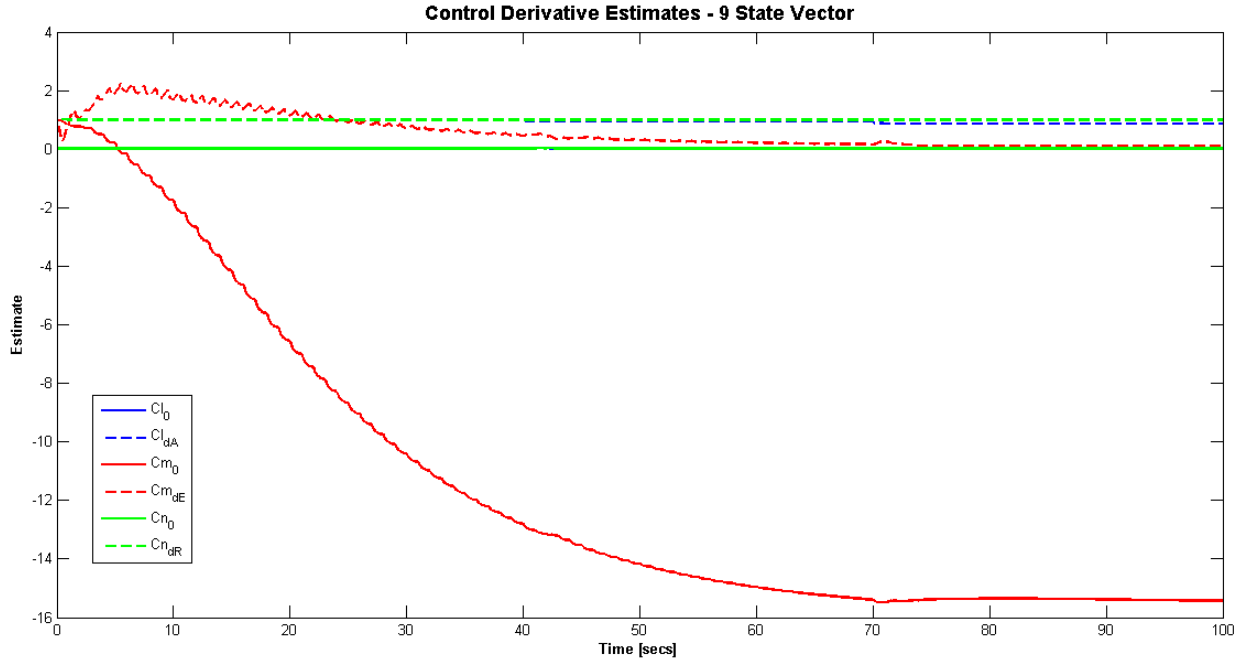

Figure 5.55: Control Derivative Estimates - 9 State Vector. Values for Cl_{dA}, Cm_{dE}, Cm_0 and Cn_{dR} have been normalised to have a value of 1, Cl_0 and Cn_0 should both be zero.

In a final attempt to solve the observability issue three separate filters were developed, one each

for roll, pitch and yaw with each filter, a 3 state filter. The states for the roll only filter are p, Cl_{dA1} and Cn_0, the pitch only filter states are q, Cm_{dE1} and Cm_0 and the yaw only filter has r, Cn_{dR1} and Cn_0 as states. The angular rate estimates are given in figure 5.56 and again show compliance with the measurements. Separating the filters caused slight improvement in the control derivative and trim estimates (figure 5.57), however the observability problem is still present particularly for the Cm_0 term. This was to be expected as the aircraft lateral dynamics have been excited by the demanded roll inputs given in figure 5.44 hence the estimates of the derivatives related to the later dynamics are more accurate than the longitudinal motion derivatives. For good estimates it is necessary to excite the aircraft dynamics.

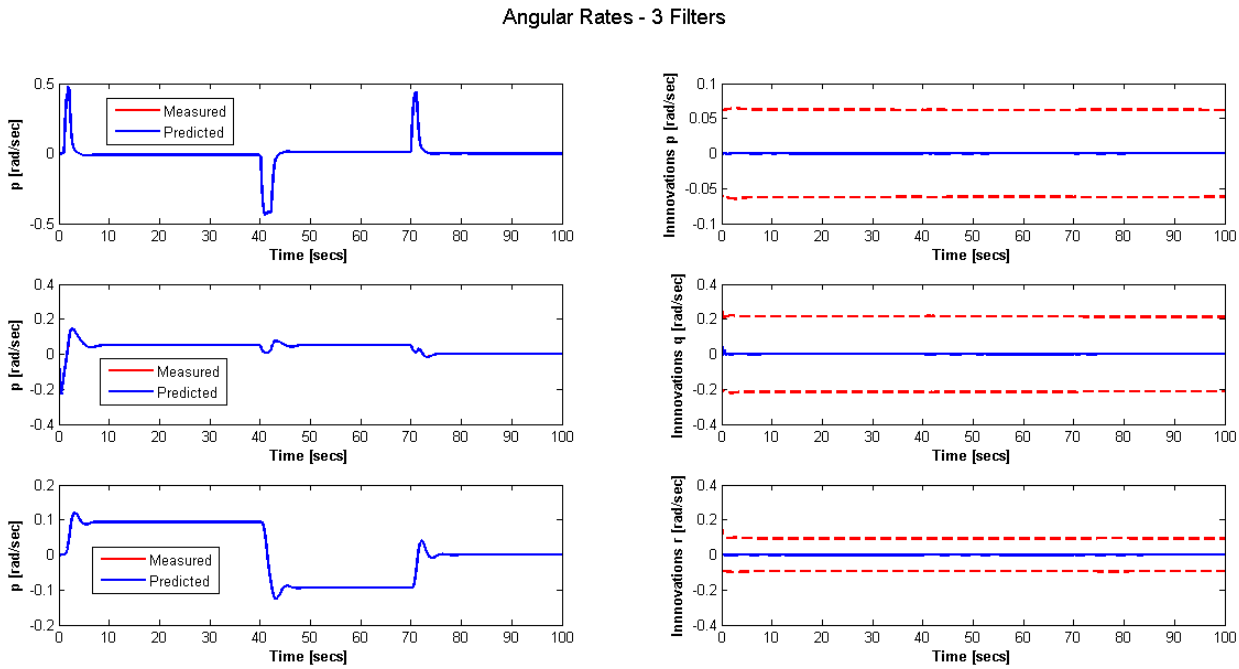

Figure 5.56: Angular Rates - 3 Filters, left column estimates (red measured, blue predicted), right column innovations (2σ uncertainty bounds (red dashed) and innovations (blue).

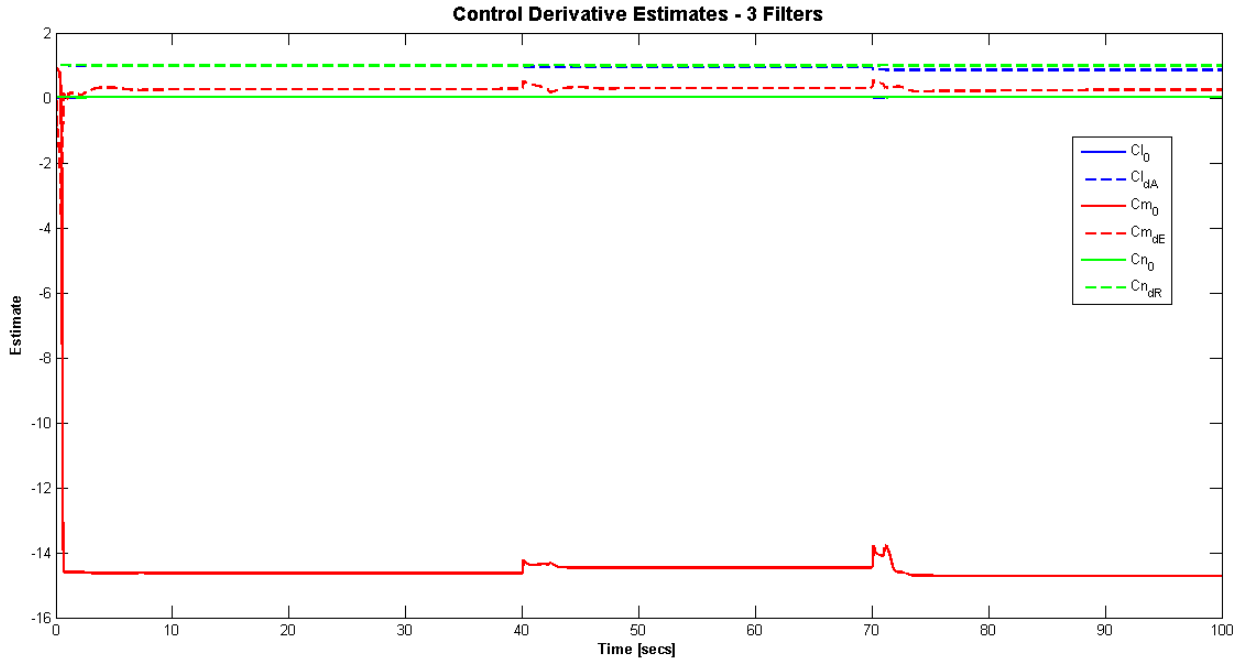

Figure 5.57: Control Derivative Estimates - 3 Filters Vector. Values for Cl_{dA}, Cm_{dE}, Cm_0 and Cn_{dR} have been normalised to have a value of 1, Cl_0 and Cn_0 should both be zero.

5.6.3 Findings

The results of section 5.5.2 illustrate that if the NMPC controller could be provided with estimates of the control derivatives it would assist the controller in allocating control authority appropriately. For this reason a UKF filter was designed in this section to provide real time estimates. However results show that many of the control derivatives are unobservable. Many attempts were made to tackle this issue, however all proved to be unsuccessful. The 3 filter solution was integrated with the NMPC controller to test the full active fault tolerant control system. Results for these tests have not been supplied as the incorrect estimates of the filter caused the solution from the controller to diverge. Further investigations into the full 6DoF fault tolerant controller are required.

As the main objective of this thesis is to demonstrate that my controller design can be utilised for fault tolerant flight control, the next section looks at the longitudinal motion of the aircraft with integrated FDI to form a full active fault tolerant flight control system.

5.7 Narrowing of Scope: Engine Failure - Loss of Power

The loss of power on an aircraft due to engine failure can result in a catastrophic breakdown of the system if left unattended. This section demonstrates the use of my active FTC system design as a fault tolerant flight controller in the event of an engine failure.

The FTC system is used to control the longitudinal motion of the aircraft. The design comprises of a UKF filter to monitor the thrust level of the air vehicle. Fault detection logic is built into the filter and once the decision is made that there is a loss of power the filter estimates are fed to the NMPC controller for reconfiguration.

The filter design is detailed in the next subsection followed by the details of the controller design. Finally numerical results are presented.

5.7.1 Filter Design

The filter design process consists of the development of a simple PID thrust controller. The NMPC and filter designs were independently constructed and tested then integrated into the final design.

5.7.1.1 PID Controller Design

Figure 5.58 is a block diagram of the inner loops developed for the filter design process. Longitudinal motion requires a height control loop and a thrust control loop, as well as a pitch control loop which calculates the amount of elevator deflection required to achieve the desired pitch angle.

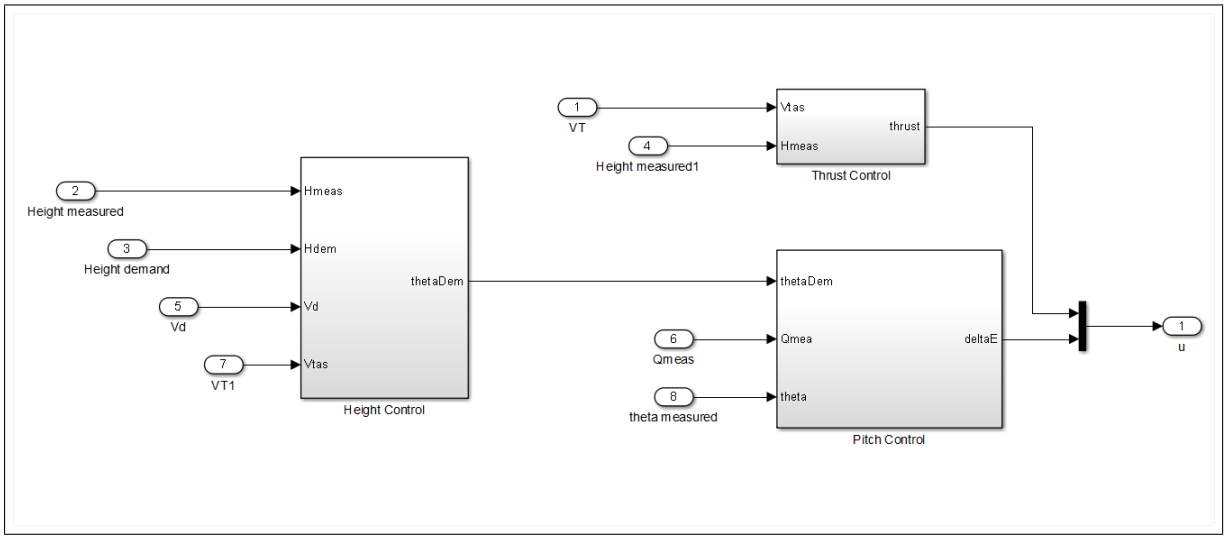

Figure 5.58: Thrust Controller Filter Design - Innerloops

The height controller is the same as that given in section 5.6.1.1 and the pitch control loop is as described in section 5.6.1.4. The thrust control loop is given in figure 5.59, and is

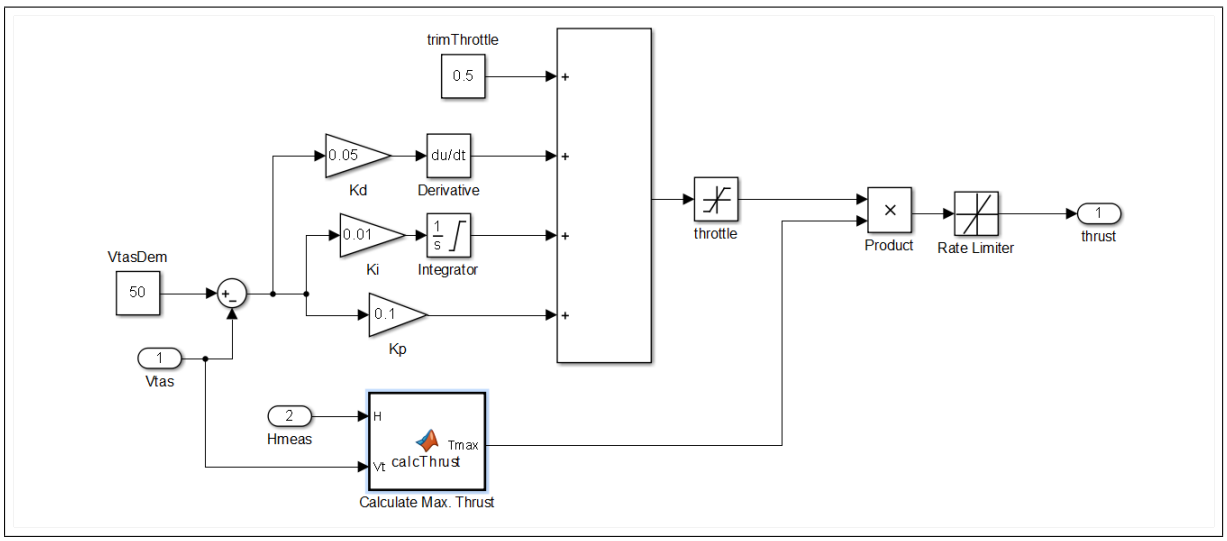

Figure 5.59: Thrust Controller Filter Design - Thrust Control Loop

very similar to the throttle control loop of section 5.6.1.2 with the addition of the "calculate maximum thrust" block. This block contains the maximum thrust expression given in equation (5.32). The maximum thrust value and the required throttle are multiplied to obtain the amount of thrust required to maintain a desired true airspeed of $50m/s$ and the demanded height.

The next subsection describes the design of the UKF filter.

5.7.1.2 UKF FDI Filter

The UKF filter is designed to estimate the amount of thrust used by the aircraft. The filter states are:

$$\mathbf{x}_{\text{ukf}} = [V_N,\ V_D,\ \theta,\ T], \tag{5.78}$$

where T is thrust. The measurement vector is:

$$\mathbf{x}_{\text{ukf}} = [V_{\text{EAS}},\ v_D,\ \theta], \tag{5.79}$$

where V_{EAS} is equivalent airspeed of the aircraft at sea level whereas V_T is the true airspeed at altitude. V_{EAS} is the speed commonly used in measurements. The relationship between V_{EAS} and V_T is given by:

$$V_{\text{EAS}} = V_T\sqrt{\frac{\rho}{\rho_0}}, \tag{5.80}$$

where ρ is the air density at a given altitude and ρ_0 is the density of air at sea level ($1.225\,\text{kg/m}^3$). For this work the aircraft is assume to be flying low enough for $V_{\text{EAS}} = V_T$.

The weighting matrices Q and R were set to:

$$\mathbf{Q} = \begin{bmatrix} (5\,\Delta t\,0.05)^2 & 0 & 0 & 0 \\ 0 & (5\,\Delta t\,0.05)^2 & 0 & 0 \\ 0 & 0 & (0.1\,\Delta\,t)^2 & 0 \\ 0 & 0 & 0 & (6500\,\Delta t\,0.3)^2 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} (0.05)^2 & 0 & 0 \\ 0 & (0.05)^2 & 0 \\ 0 & 0 & (0.017)^2 \end{bmatrix}, \tag{5.81}$$

where Δt is the filter update rate 0.01 secs. The intial state vector and covariance matrix are:

$$\mathbf{x}(0) = [50,\ 0\ 0.04247,\ 1507.7526]^\intercal, \quad \mathbf{P}(0) = \begin{bmatrix} (0.5)^2 & 0 & 0 & 0 \\ 0 & (0.5)^2 & 0 & 0 \\ 0 & 0 & (0.017)^2 & 0 \\ 0 & 0 & 0 & (315)^2 \end{bmatrix} \tag{5.82}$$

5.7.1.3 Numerical Results

The following test cases were carried out to examine the filter performance:

Test 1: no fault,

Test 2: 70% loss of power 70 secs into flight,

Test 3: 90% loss of power 35 secs into flight,

Test 4: 50% loss of power 20 secs into flight.

The aircraft was required to fly the trajectory given in figure 5.5. The effects of wind and turbulence have been taken into account as well as the effect of noise on the measurements of V_{EAS}, v_d and θ, which has been modelled as a normally distributed random white noise.

To analyse the performance of the filter the innovation covariance plots were examined and are given in figures 5.60, 5.61 and 5.62 for V_{EAS}, v_d and θ respectively. The results show that for all test cases the innovations are well within the 2σ covariance bounds. The test case 3 where the thrust level drops to 10% shows that after approximately 70 seconds the aircraft is unable to maintain flight as there is not enough power hence the filter diverges. The thrust estimates are given in figure 5.63 along with the actual thrust applied to the aircraft. In each test case the filter does an excellent job of estimating the thrust levels.

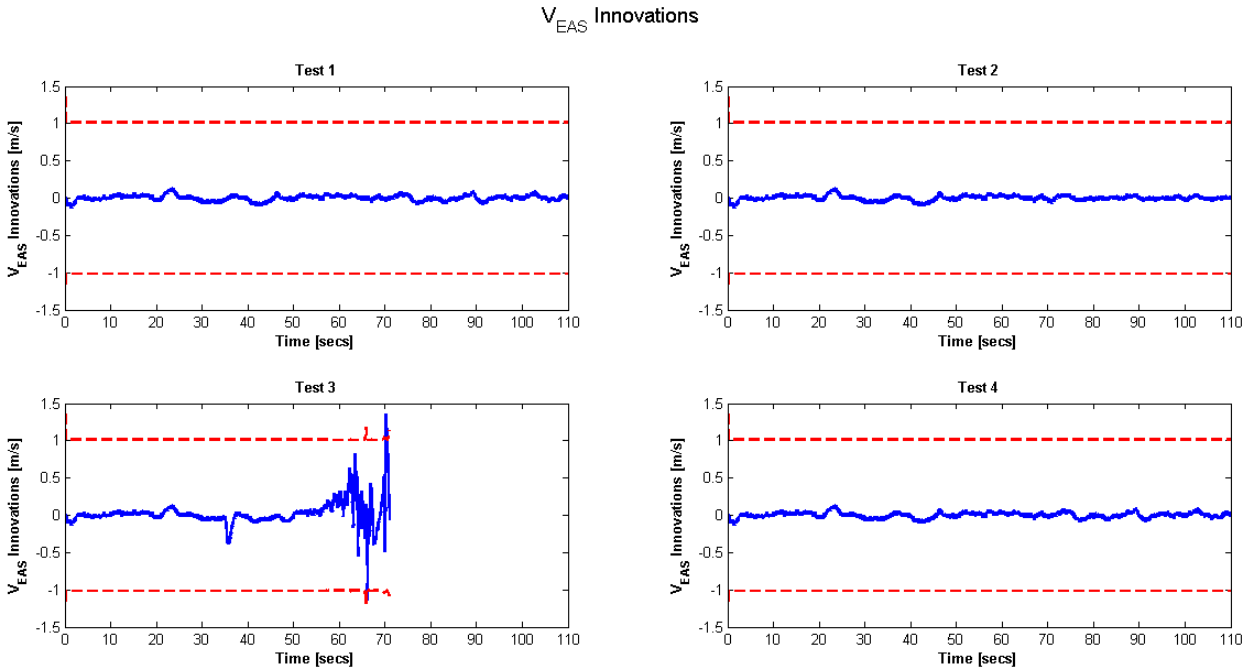

Figure 5.60: UKF V_{EAS} Innovations - Longitudinal Model, $\pm 2\sigma$ innovation covariance bounds (red dashed lines), V_{EAS} innovations (solid blue line)

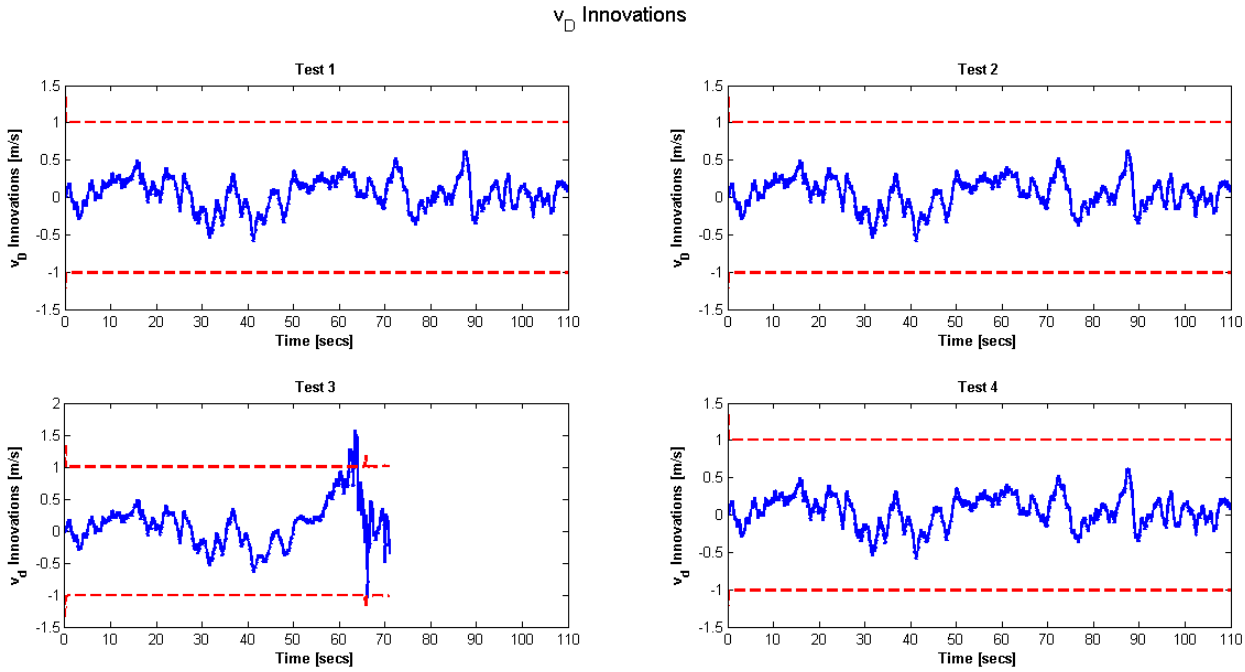

Figure 5.61: UKF V_D Innovations - Longitudinal Model, $\pm 2\sigma$ innovation covariance bounds (red dashed lines), V_D innovations (solid blue line)

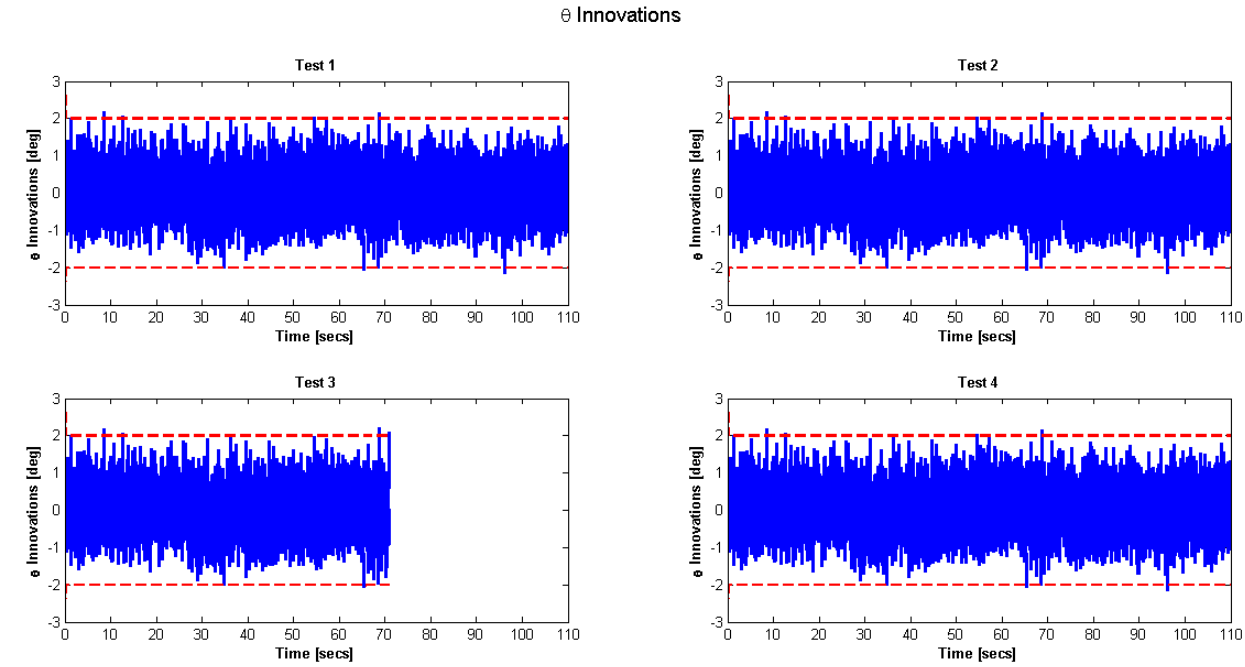

Figure 5.62: UKF θ Innovations - Longitudinal Model, $\pm 2\sigma$ innovation covariance bounds (red dashed lines), θ innovations (solid blue line)

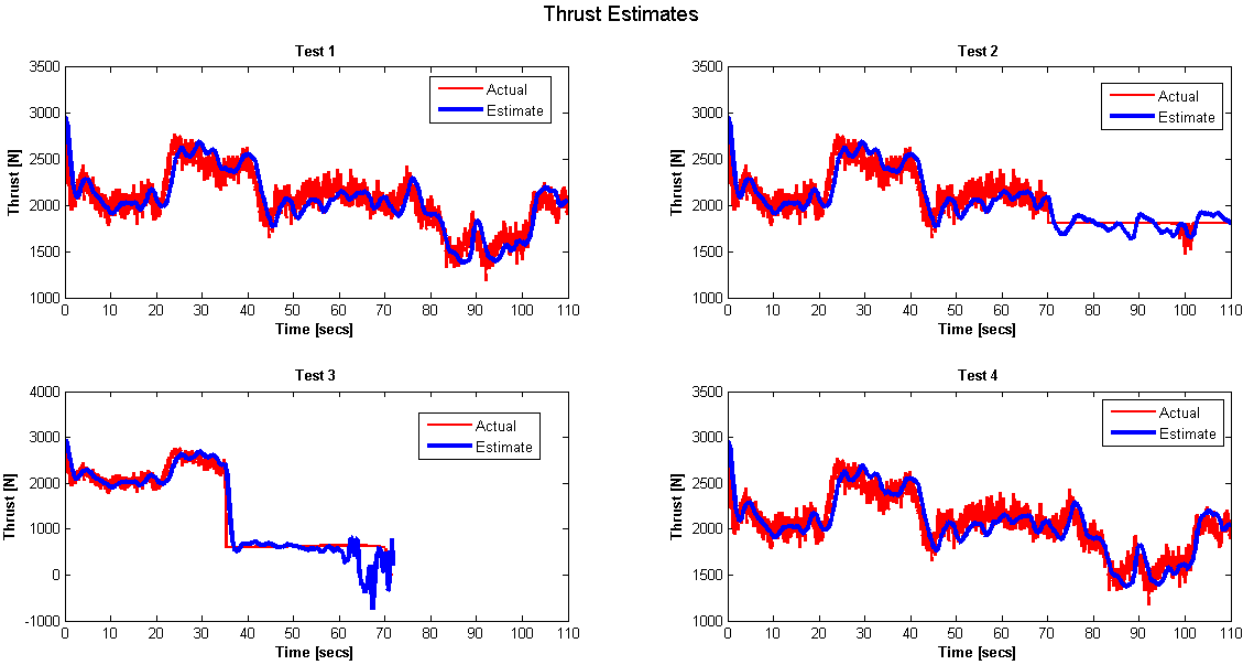

Figure 5.63: UKF Thrust Estimates - Longitudinal Model

5.7.1.4 Fault Detection Logic

The premise behind the FTC design is the provision of the updates of the power status to the NMPC controller to enable controller reconfiguration. When an engine fails the amount of thrust available decreases. If this level of thrust could be estimated and provided to the NMPC controller the maximum constraint on thrust can be updated and the controller can then allocate control authority to the control inputs accordingly. For this reason it is important to detect the fault and to know when to begin feeding the controller with filter estimates of thrust, hence the need for fault detection logic.

The controller is designed (section 5.7.2) to calculate the optimal amount of thrust to maintain a height demand and true airspeed. The filter on the other hand estimates the thrust level currently used by the aircraft, hence if the demand is greater than the estimate this would indicate an engine failure.

The fault detection logic therefore checks whether the thrust demand is higher than the thrust estimate. If this is true for a set period of time then a fault has occurred and a flag is turned on, indicating that a fault has occurred, and consequently the constraints must be updated via the filter estimates. The filter outputs an estimate of the state as well as the uncertainty on the estimate, so the actual value of the state as predicted by the filter is within plus or minus the uncertainty. For this reason a number of tests were performed to see whether the check should include zero level of uncertainty, $\pm 1\sigma$ uncertainty or $\pm 2\sigma$, and the results are given in figure 5.64. The results are based on a fault count of 200, i.e. when the demand is greater than the thrust estimate the fault counter is incremented by one, and when this counter exceeds 200 the fault flag switches from 0 to 1 indicating to the controller that the maximum constraint on thrust must be updated with the filter estimate. The number of counts being set to 200 is based purely on trial and error. The results show that the filter estimate plus 2σ uncertainty was able to correctly identify the fault within approximately a couple of seconds of the fault occurring. The other uncertainty bounds as well as the zero uncertainty case all indicated false detection of the fault, that is the fault flag is set to true at the incorrect times. Note that a fault was not detected for test case 4 even with a 2σ uncertainty bound. This is because the thrust estimate plots (as shown in figure 5.63) indicate that in a no fault case the aircraft requires no more than 50% of the maximum thrust to maintain the given trajectory, hence the demand is at all times less than the estimate.

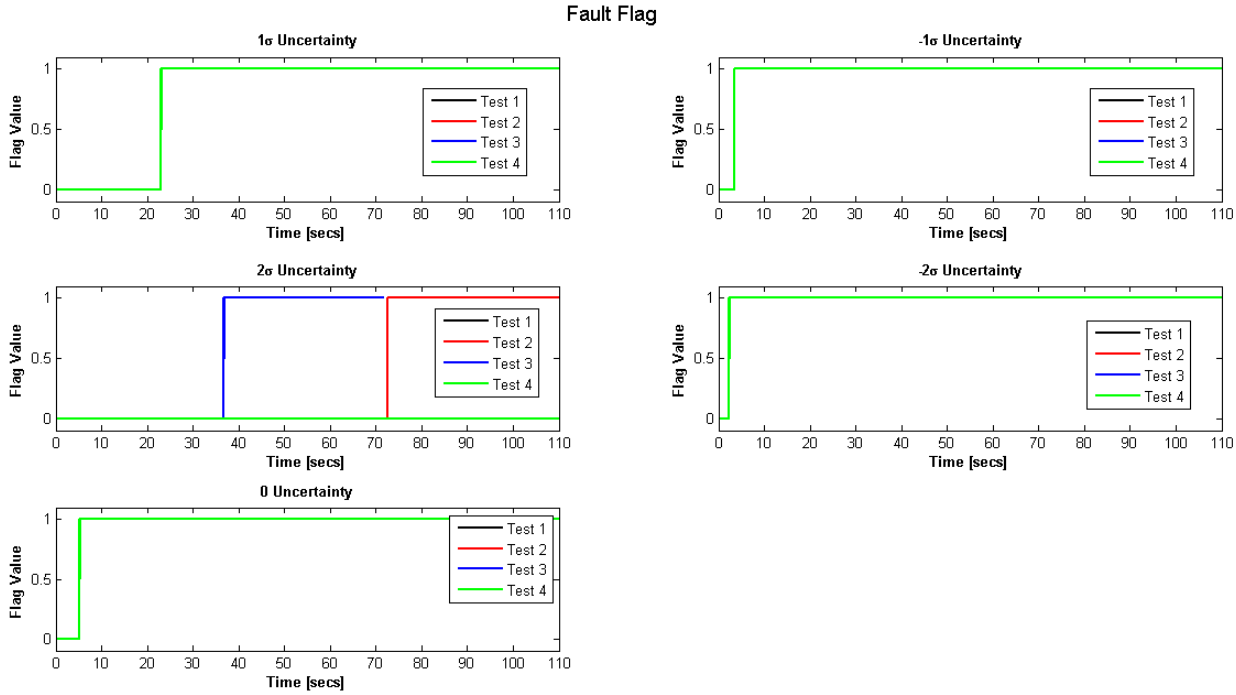

Figure 5.64: UKF Fault Flag - Longitudinal Model

In the next section the complete active FTC system design for thrust control is detailed.

5.7.2 Controller Design

This section steps through the controller design for the active FTC system for the longitudinal motion of the aircraft.

Pseudospectral discretisation is applied to the controller design and the NMPC state vector is:

$$\mathbf{x_{nmpc}} = [x_D,\ V_N,\ V_D,\ \theta,\ q,\ \delta_{\text{thrust}},\ \delta_e,\ \Delta\delta_{\text{thrust}}\ \Delta\delta_e]^\intercal\,, \tag{5.83}$$

The following optimal control problem is solved each time step:

$$\begin{aligned}\min_{\mathbf{x},\mathbf{u}}\ \frac{H_p}{2}\sum_{j=1}^{j=N+1}\Big(&\left\|\mathbf{x}_D(j)-\mathbf{x}_{D_{\text{ref}}}(j)\right\|^2_{Q_x}+\left\|\mathbf{V}_T(j)-\mathbf{V}_{T_{\text{ref}}}(j)\right\|^2_{Q_{VT}}\\&+\left\|\mathbf{V}_D(j)-\mathbf{V}_{D_{\text{ref}}}(j)\right\|^2_{Q_{VD}}+\left\|\Delta\delta_{\text{thrust}}\right\|^2_{Q_T}+\left\|\Delta\delta_e\right\|^2_{Q_{\delta_e}}\Big)\,w(j),\end{aligned} \tag{5.84}$$

subject to

$$\left(\frac{t_f - t_0}{2}\right)\mathbf{D}_{j,k}\mathbf{x}\left(t\right) - \dot{\mathbf{x}} = 0, \tag{5.85}$$

$$\mathbf{x}(t_0) - \mathbf{x}_{\text{dem}}(t_0) = 0, \tag{5.86}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{5.87}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}, \tag{5.88}$$

$$\Delta\mathbf{u}_{\text{lb}} \leq \Delta\mathbf{u} \leq \Delta\mathbf{u}_{\text{ub}}, \tag{5.89}$$

where V_T and $V_{T_{\text{ref}}}$ are the actual and reference true airspeeds respectively and Q_x, Q_{VT}, Q_{VD}, Q_T and Q_{δ_e} are weighting parameters and the term $w(j)$ are the pseudospectral node weights as defined in chapter 3. The constraints applied are given in Table 5.7. The cost function weights are square matrices with the following diagonal values for each state: $Q_x = 3$, $Q_{VT} = 3$, $Q_{VD} = 5$, $Q_T = 0.001$ and $Q_{\delta_e} = 0.1$.

Table 5.7: Constraints for Longitudinal Motion, Thrust Controller

Variable	**Upper Constraint**	**Lower Constraint**
x_D	300 m	1 m
V_N	100 m/s	30 m/s
V_D	3 m/s	−3 m/s
θ	None	None
q	None	None
δ_e	20 deg	−20 deg
$\Delta\delta_{\text{thrust}}$	6500 N/s	−6500 N/s
$\Delta\delta_e$	200 deg/s	−200 deg/s

The lower limit on thrust is 0 N while the upper limit changes throughout the flight and is set to the maximum value of thrust based on the height of the aircraft. Maximum thrust is calculated via equation (5.32). If a fault has been detected and the fault flag of section 5.7.1.4 is set to 1 the upper constraint is set to the filter estimate of thrust plus a 2σ uncertainty.

The following scenarios were designed to test the fault tolerant control system:

Scenario 1: no fault case

Scenario 2: engine failure - 65% power loss 30 secs into flight,

Scenario 3: engine failure - 70% power loss 30 secs into flight.

Note: all test runs take into account the effect of wind.

Figures 5.65, 5.66 and 5.67 show the control inputs for scenarios 1, 2 and 3 respectively. The results for the thrust show a dip in the constraint value for the upper thrust limits for scenarios 2 and 3 just after 30 secs. This indicates that the fault was correctly identified and the NMPC was reconfigured with the information provided by the FDI filter. The uncertainty bounds in both figures are slightly higher than the actual thrust applied due to the addition of the 2σ uncertainty. Other values of σ were found to cause the controller and hence the filter to diverge. Although the estimate is slightly above the actual it is still in the vicinity of the actual thrust level and prompts the controller to allocate more control authority to the other available actuators. The results show that compared to the no fault case once a fault occurs the elevator activity increases as the power decreases. Also the more severe the fault the faster the detection time. This is evident from the fact that the fault is detected earlier in the 70% power loss case compared to the 65% loss of power case.

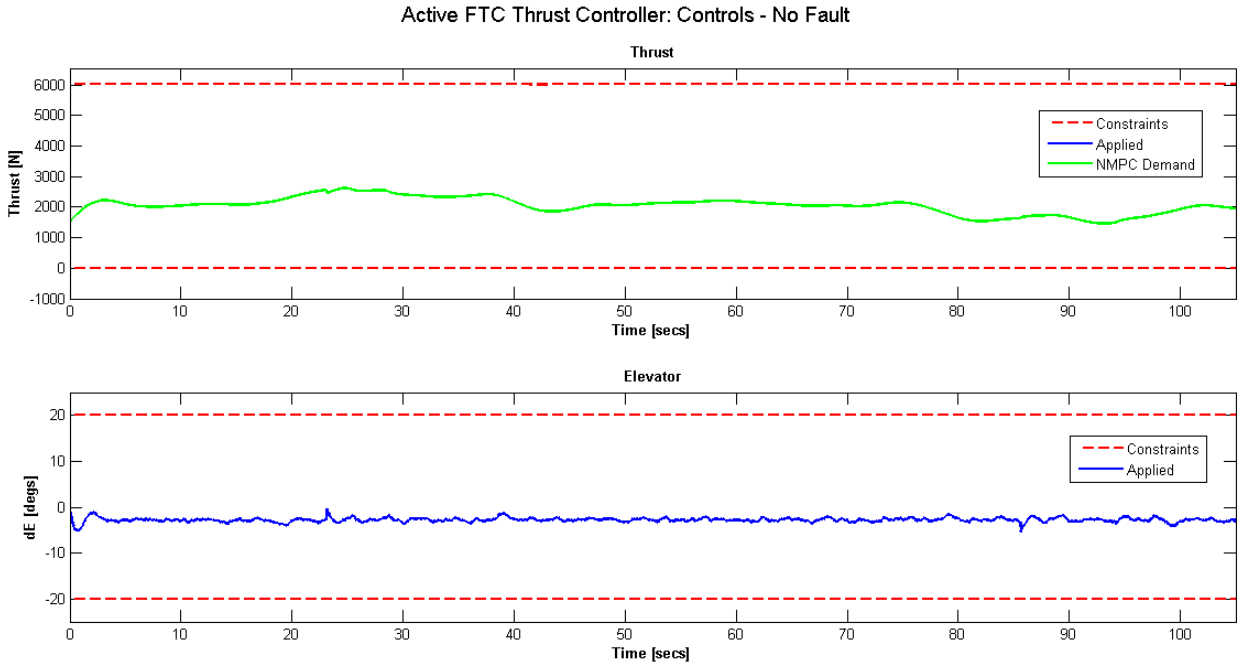

Figure 5.65: Active FTC Thrust Controller: Control Inputs - Scenario 1: No Fault Case

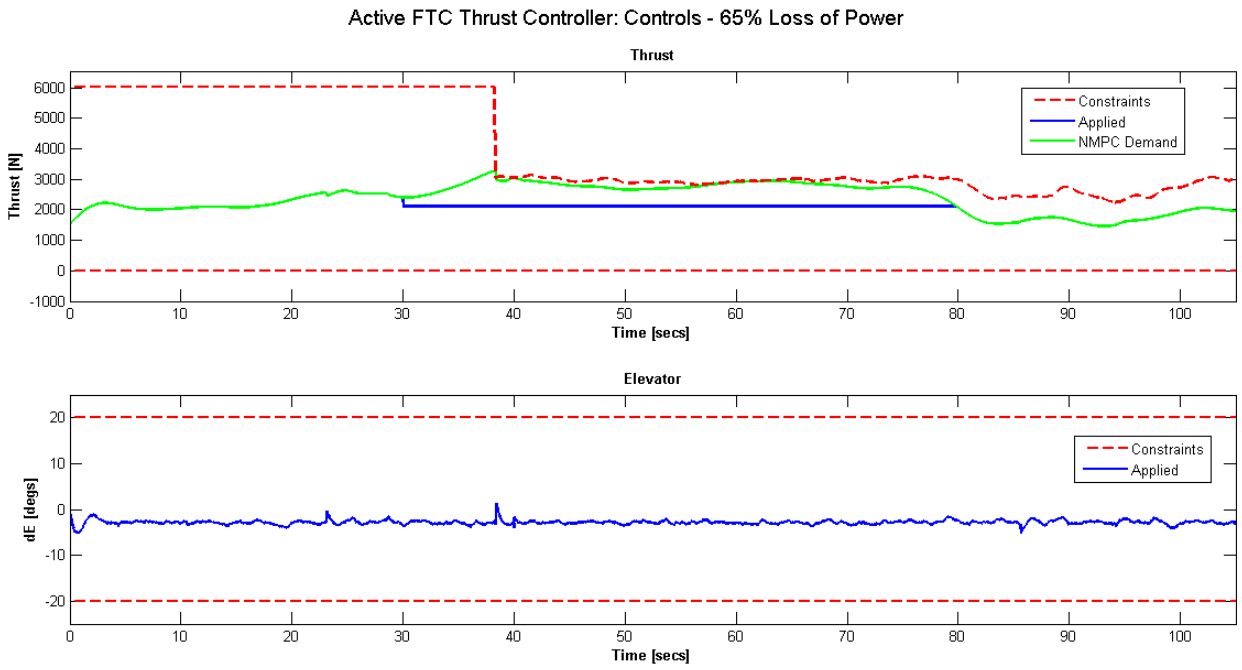

Figure 5.66: Active FTC Thrust Controller: Control Inputs - Scenario 2: 65% Loss of Power Case

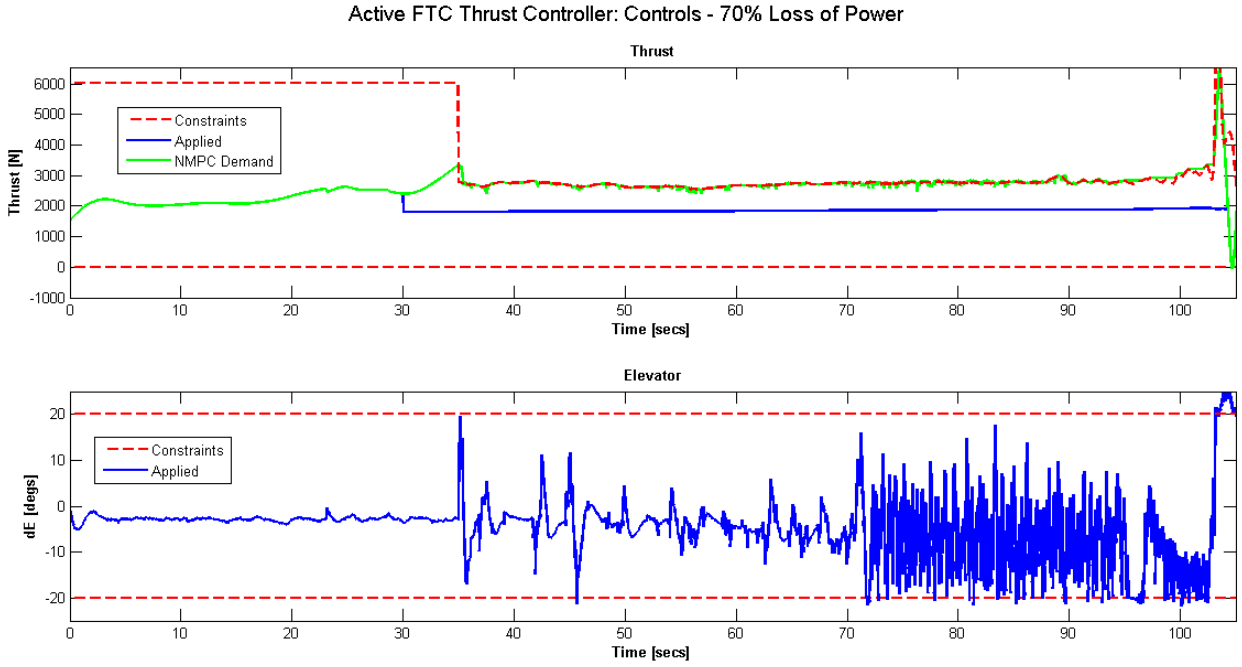

Figure 5.67: Active FTC Thrust Controller: Control Inputs - Scenario 3: 70% Loss of Power Case

Figure 5.68 shows that true airspeed of the air vehicle. A true airspeed of 50m/s was demanded by the aircraft. In the case where there is 65% loss of power the the aircraft is unable to maintain

the demanded true airspeed during straight and level flight. Once the aircraft begins to descend the demanded true airspeed is recovered. However in the case of 70% power loss there is not enough power to maintain the demanded airspeed. Once the fault occurs the airspeed begins to drop and reaches the stall speed (approximately $21m/s$) causing the aircraft to lose control.

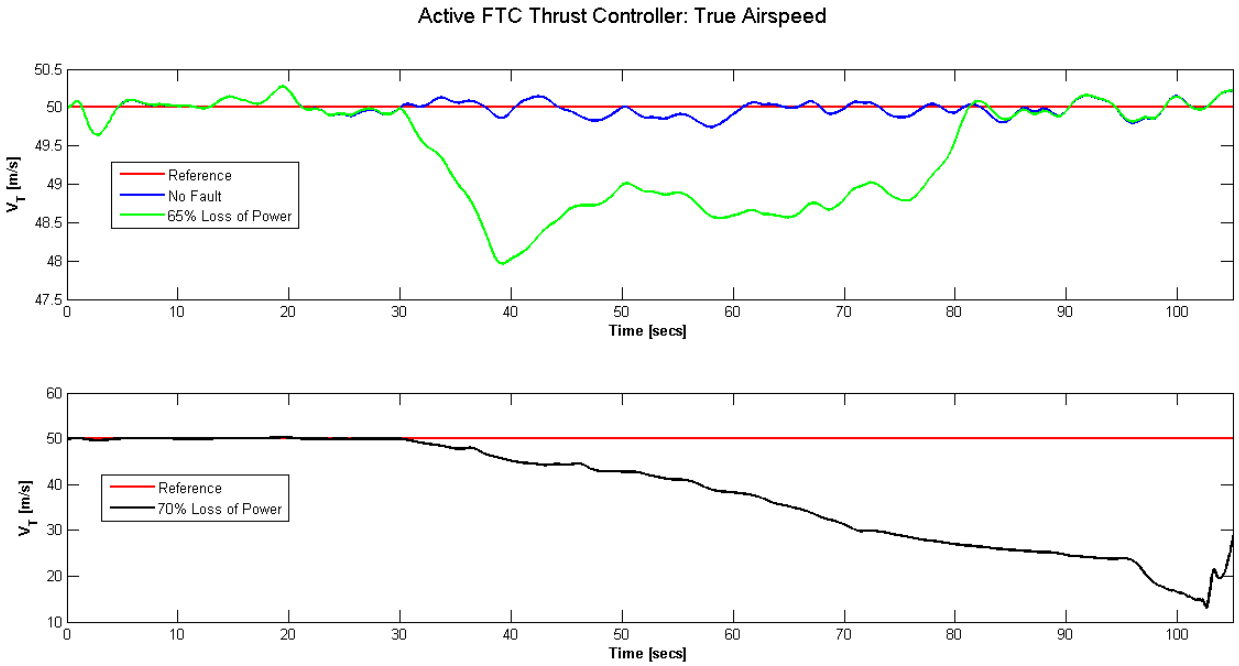

Figure 5.68: Active FTC Thrust Controller: True Airspeeds

The pitch angle, pitch rate, AoA and the g-forces experienced by the aircraft are shown in figures 5.69, 5.70, 5.71 and 5.72 respectively. The AoA 15 deg limit was only exceeded in the 70% loss of power case meaning during this time the equations of motion were invalid. The maximum g-force once again was seen to be approximately 1.5g's in all cases however in the 70% loss of power case the maximum reached was 2g's. Overall the first priority is to save the aircraft even if it means the aircraft is be pushed to its mechanical and structure limits.

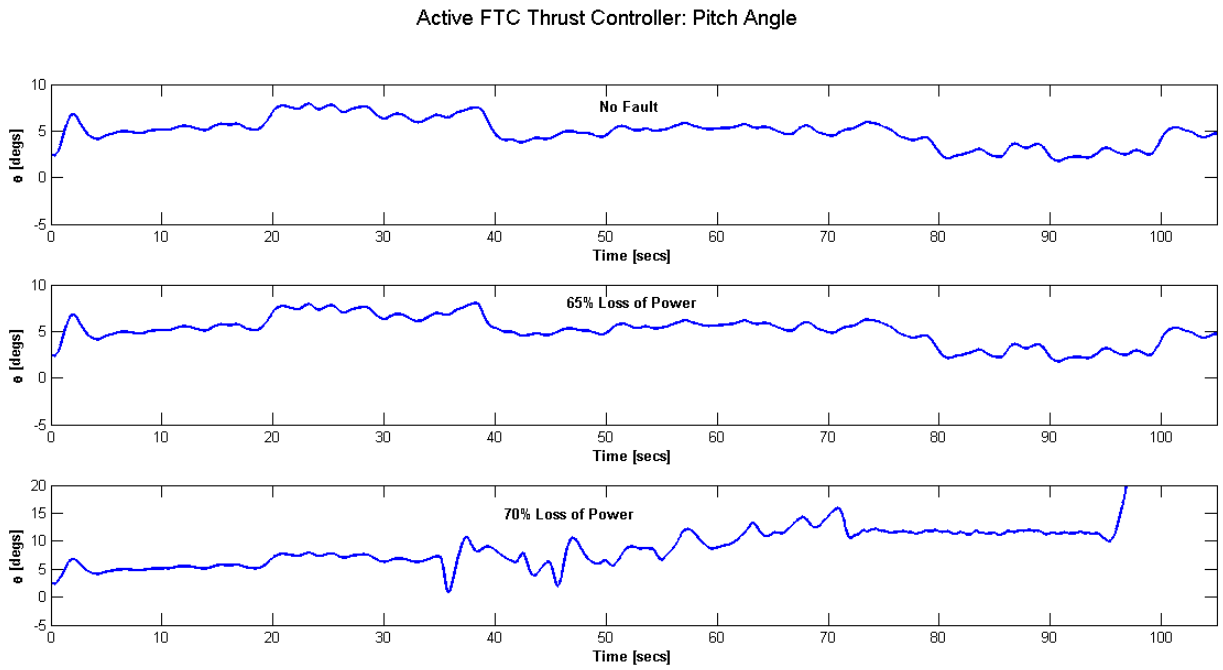

Figure 5.69: Active FTC Thrust Controller: Pitch Angle, θ

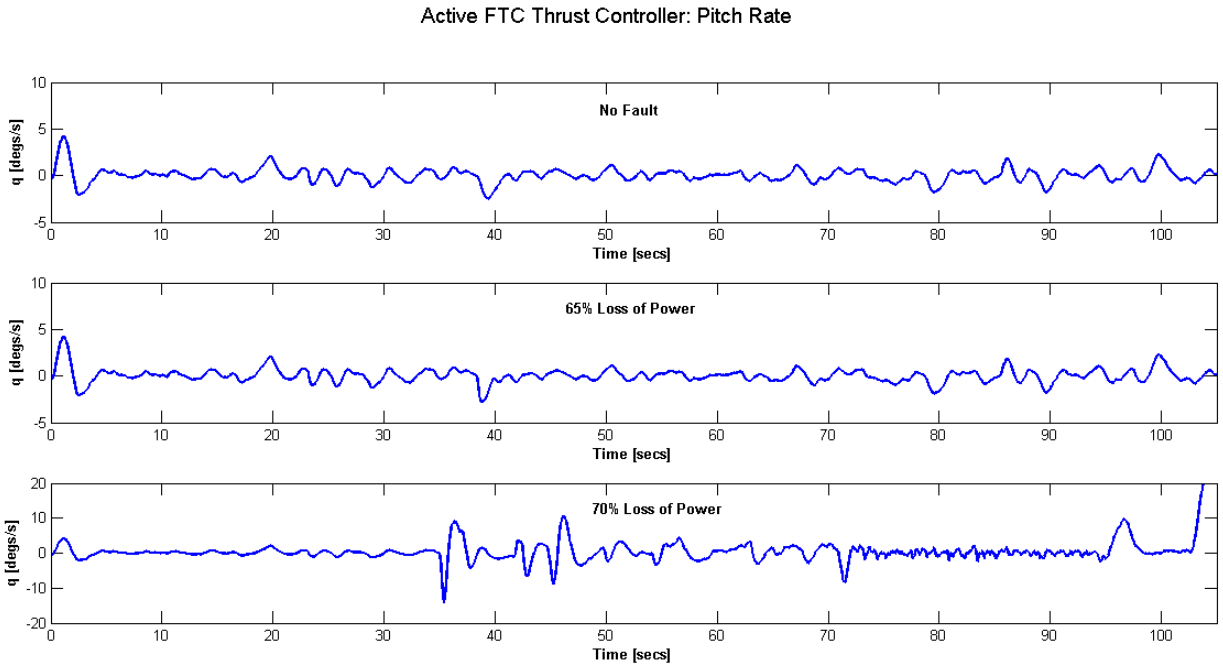

Figure 5.70: Active FTC Thrust Controller: Pitch Angle, q

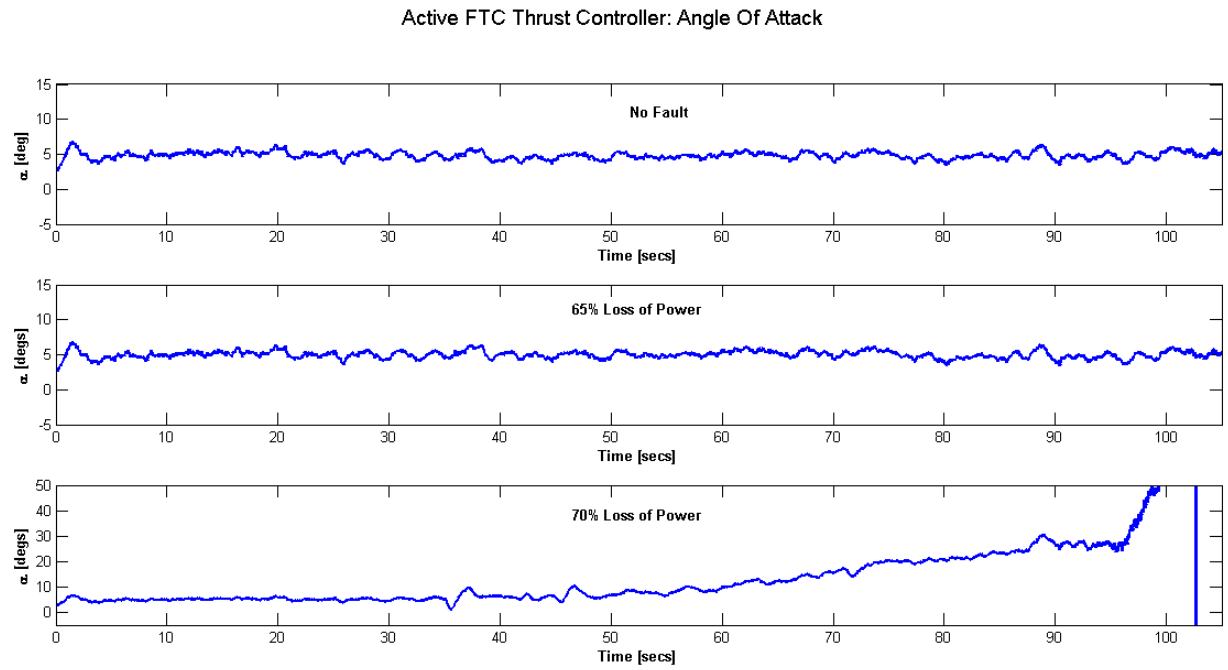

Figure 5.71: Active FTC Thrust Controller: Angle of Attack, q

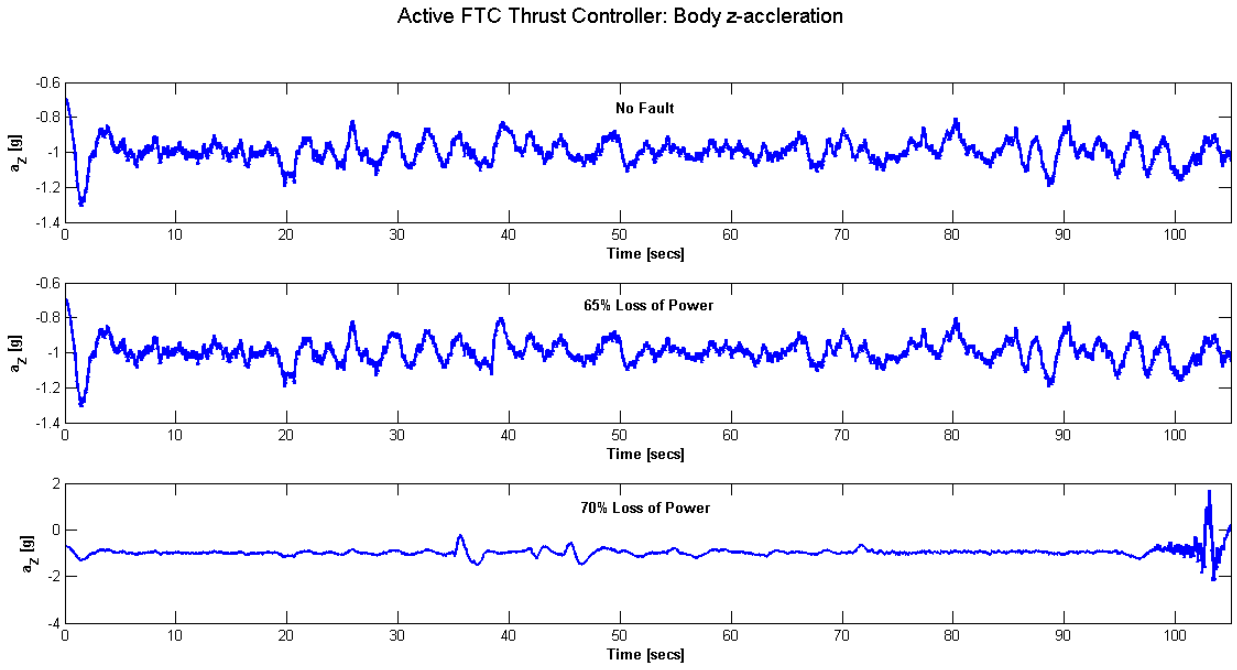

Figure 5.72: Active FTC Thrust Controller: g-Forces, Body Acceleration a_Z

The vertical speed (also known as climb rate) response is given in figure 5.73. In the 65% power loss case the response is very similar to the no fault response. A 70% loss in power results in the aircraft being unable to maintain speed and it descends to the ground.

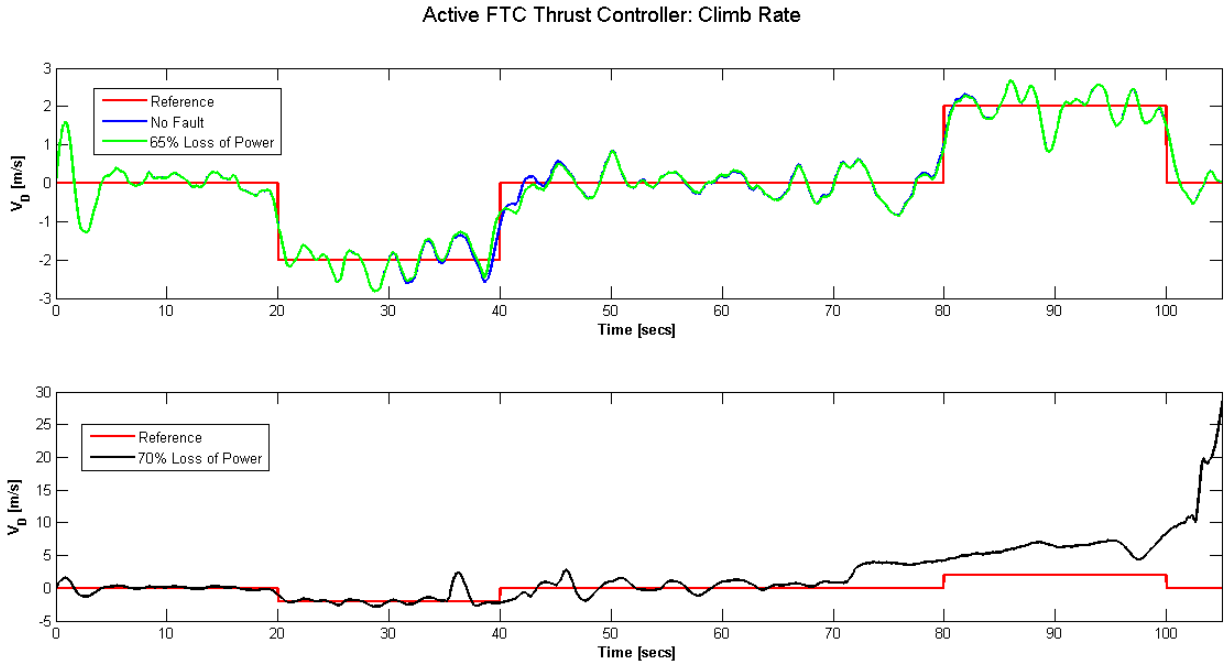

Figure 5.73: Active FTC Thrust Controller: Climb Rates

The height profiles given in figure 5.74 show that even with a 65% loss in power the aircraft is capable of maintaining the reference trajectory. However when the power decreases by another 5% the aircraft completes the climb to the highest demanded altitude but begins to descend half way through straight and level flight.

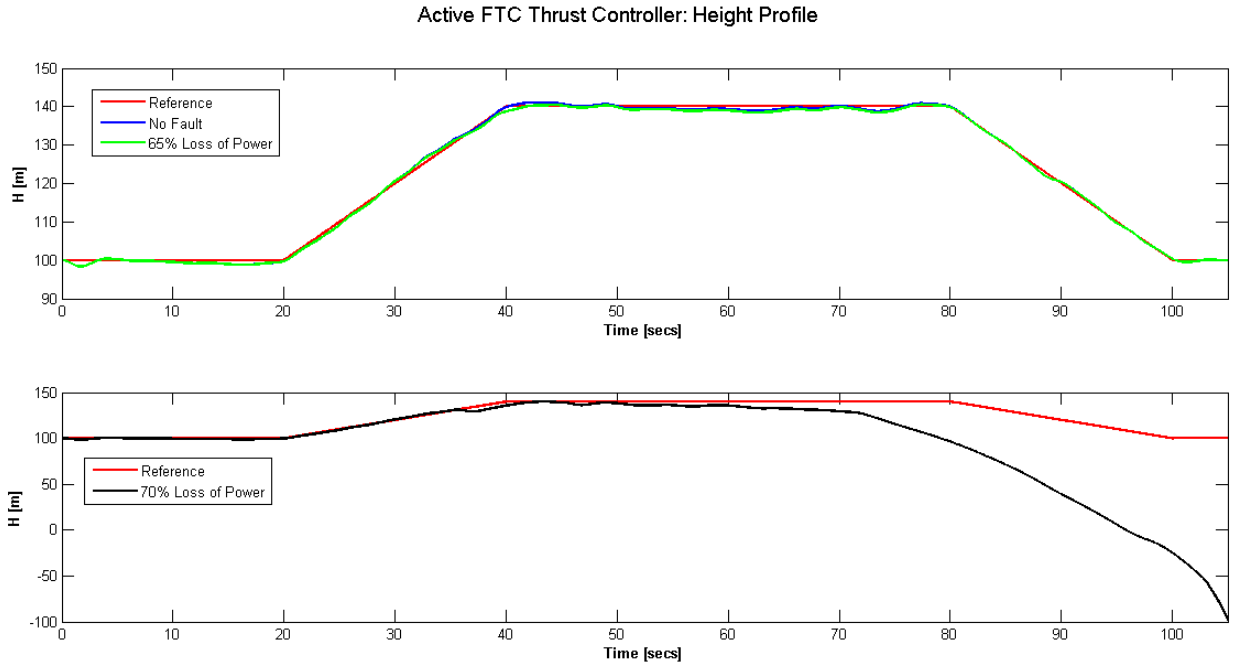

Figure 5.74: Active FTC Thrust Controller: Height Profiles

The results obtained successfully demonstrate the application of my active fault tolerant flight control system design. The control system is able to detect an engine fault within 2-3 seconds of the fault occurring enabling reconfiguration of the NMPC controller to allow reallocation of control authority to maintain the aircraft on the demanded flight path within the aircraft limits. The controller works hard to achieve the demands, however in the event where this is impossible this information can be used to bring the aircraft back safely to the ground.

5.8 Findings and Conclusion

The results of this chapter have shown the successful application of my active fault tolerant control system design from chapter 4 to flight control. In this chapter a generic aircraft model was developed and the active FTC system was applied to the 3DoF aircraft model. Research into the application of the FTC system on a full 6Dof model was also conducted however many problems were encountered particularly in the design of the FDI system. This has been identified as further research as it is deemed beyond the scope of this thesis.

Chapter 6

UAV Case Study

6.1 Introduction

6.1.1 Motivation

Chapter 5 demonstrated the feasibility of my fault tolerant control (FTC) system design for fault tolerant flight control with the system being applied to a generic aircraft model. The main aim of this thesis is the design of a fault tolerant flight control system for an unmanned aerial vehicle (UAV). This chapter investigates the integration of the controller design into an actual UAV model.

6.1.2 Outline

The chapter begins with summarising the details of the UAV model (section 6.2). This is followed by a in depth look at the development of the nonlinear model predictive control (NMPC) controller in section 6.3. Finally the active FTC system developed in section 5.7 is applied to the UAV model in section 6.4 and its effectiveness is demonstrated through investigation of a number of different scenarios. The chapter concludes with a brief summary of findings (section 6.5).

6.2 Aircraft Details

The previous chapter developed the aircraft prediction model for the NMPC controller using a generic fictional aircraft model based on the McDonnell Douglas F-4 aircraft [52]. The model used in this section is of an actual UAV currently in operation [128]. The model is based on real flight data and comes with a confidentiality condition attached to it hence many of the aircraft

details cannot be disclosed. The UAV model is of a twin engine, propeller driven aircraft with dimensions as given in Table 6.1.

Table 6.1: UAV Data

Wingspan, b	$5.5\,m$
Chord, c	$0.55\,m$
Wing Area, S	$3\,m^2$
Mass, m	$36.8\,\text{kg}$
Propeller Diameter, D	$0.4572\,m$
Stall Speed, V_{stall}	$12\,m/s$

The control inputs used to fly the aircraft include the throttle, aileron, elevator, rudder and flaps. The thrust controller of section 5.7 will be applied to the model hence only longitudinal motion is considered and consequently, only the elevator and throttle inputs are required.

The next section provides details of the development of the NMPC controller for the given UAV model.

6.3 NMPC Controller Development

The UAV model provided uses experimental data with a series of lookup tables incorporated to find force and moment coefficients. This model will be the plant model for this part of the investigation. For the NMPC prediction model, on the other hand, an approximation model needs to be developed that will use mathematical expressions to find the force and moment coefficients rather than look up tables. Before the NMPC controller can be developed, the approximation model must be produced, tested and validated and then the NMPC controller can be designed.

I start with the development of the approximation model for the given aircraft in the next subsection.

6.3.1 Development of Approximation Model

The first step in building the approximation model is to fit polynomial curves to the experimental aerodynamic data. The aerodynamic data includes lift force coefficient CL, drag force coefficient

CD and pitching moment coefficient CM all of which are given as functions of the angle of attack α. The plots given in figures 6.1, 6.2 and 6.3 show the curve fits for CL, CD and CM respectively.

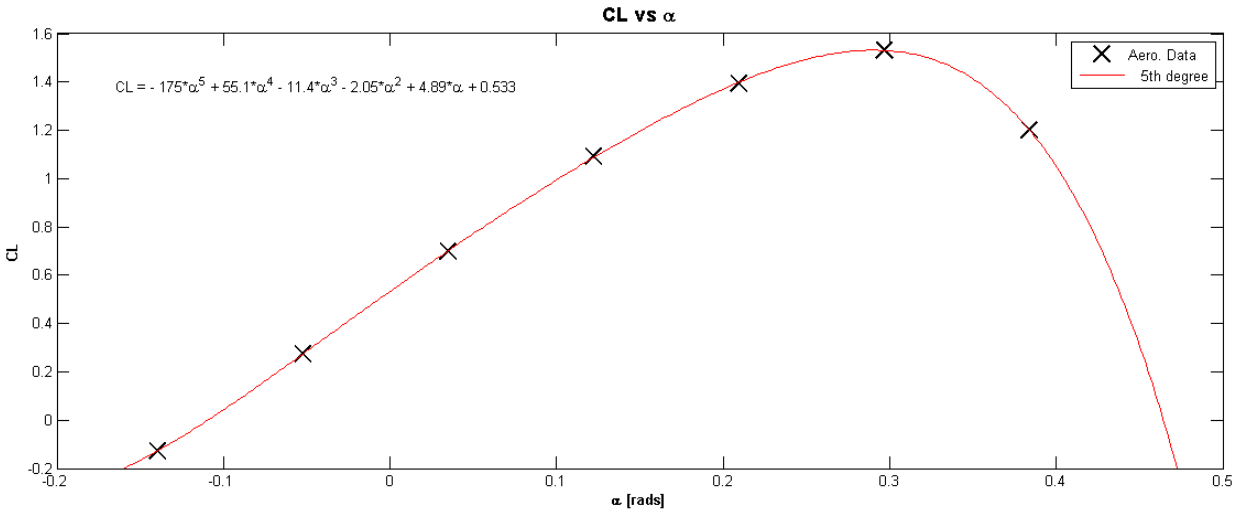

Figure 6.1: Curve Fitting to CL Data, experimental data (X), curve fit (red line)

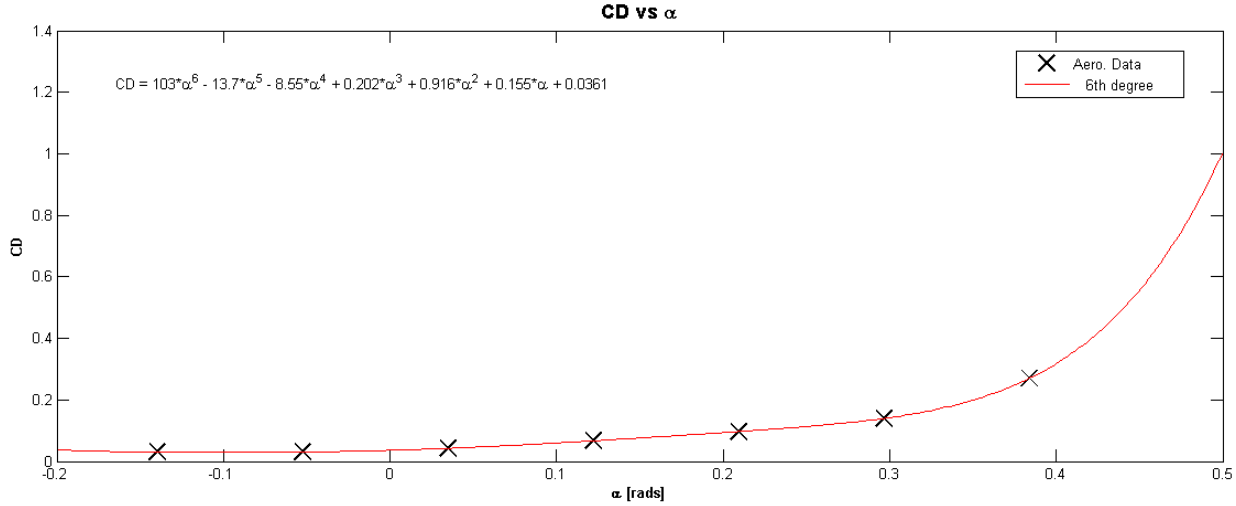

Figure 6.2: Curve Fitting to CD Data, experimental data (X), curve fit (red line)

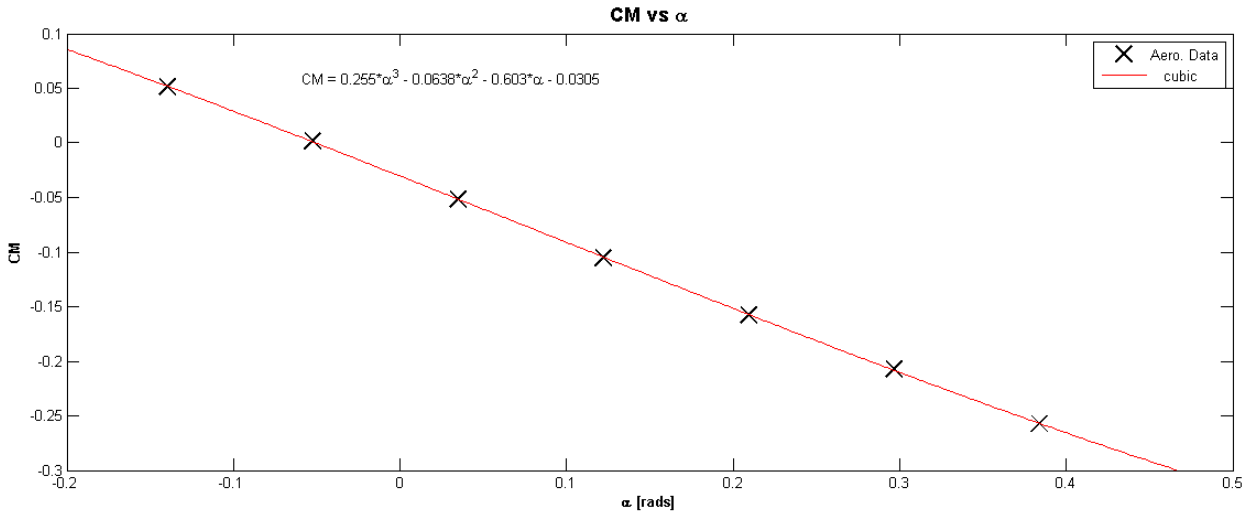

Figure 6.3: Curve Fitting to CM Data, experimental data (X), curve fit (red line)

A fifth order polynomial was fitted to the CL data using the MATLAB curve fitting toolbox (see figure 6.1). The order of the polynomial is based on trial and error to find the order of best fit. The following expression was found for CL as a function of α:

$$\mathrm{CL}(\alpha) = -175\,\alpha^5 + 55.1\,\alpha^4 - 11.4\,\alpha^3 - 2.05\,\alpha^2 + 4.89\,\alpha + 0.533. \tag{6.1}$$

Using the same procedure, sixth order and cubic polynomials were fitted to the CD and CM data respectively (figures 6.2 and 6.3 respectively). The following expressions for CD and CM were found as functions of α:

$$\mathrm{CD}(\alpha) = 103\,\alpha^6 - 13.7\,\alpha^5 - 8.55\,\alpha^4 + 0.202\,\alpha^3 + 0.916\,\alpha^2 + 0.155\,\alpha + 0.0361, \tag{6.2}$$

$$\mathrm{CM}(\alpha) = 0.255\,\alpha^3 - 0.0638\,\alpha^2 - 0.603\alpha\ - 0.0305. \tag{6.3}$$

Experimental data was also provided for the engines. To calculate the thrust force coefficient polynomials were fitted to the angular speed of the propeller (ω_{prop} measured in [rads/sec]) vs throttle (figure 6.4).

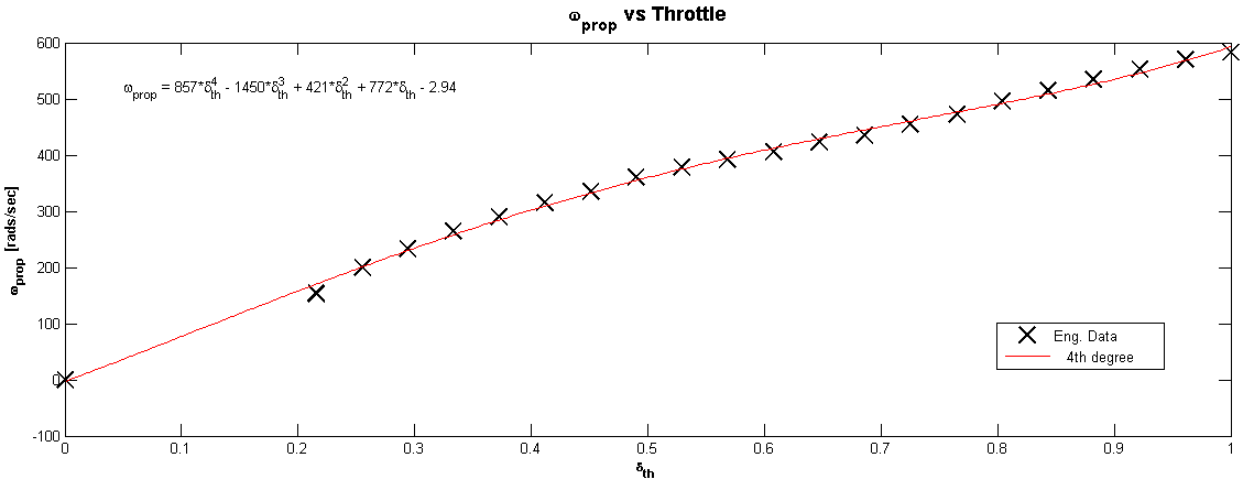

Figure 6.4: Curve Fitting to ω_{prop} Data, experimental data (X), curve fit (red line)

The following expression for ω_{prop} as a function of δ_{th} was found:

$$\omega_{\text{prop}}(\delta_{\text{th}}) = 857\,\delta_{\text{th}}^4 - 1450\,\delta_{\text{th}}^3 + 421\,\delta_{\text{th}}^2 + 772\,\delta_{\text{th}} - 2.94. \tag{6.4}$$

Given an ω_{prop} value, the advance ratio, J, of the propeller can be determined from equation (6.5):

$$\text{J} = \frac{V_T}{n\,D}, \tag{6.5}$$

where n is the rotational speed of the propeller given in revolutions per second [RPS]:

$$n = \frac{\omega_{\text{prop}}}{2\pi}. \tag{6.6}$$

ω_{prop} is the rotational speed of the propeller give in [rads/sec] calculated by equation (6.4), D is the propeller diameter and V_T is the true airspeed of the aircraft. The advance ratio, J, is the non-dimensional parameter used to describe the incoming angle of the fluid relative to the propeller blade. Using experimental data, the thrust coefficient, CT was mapped as a function of J, as given in figure 6.5.

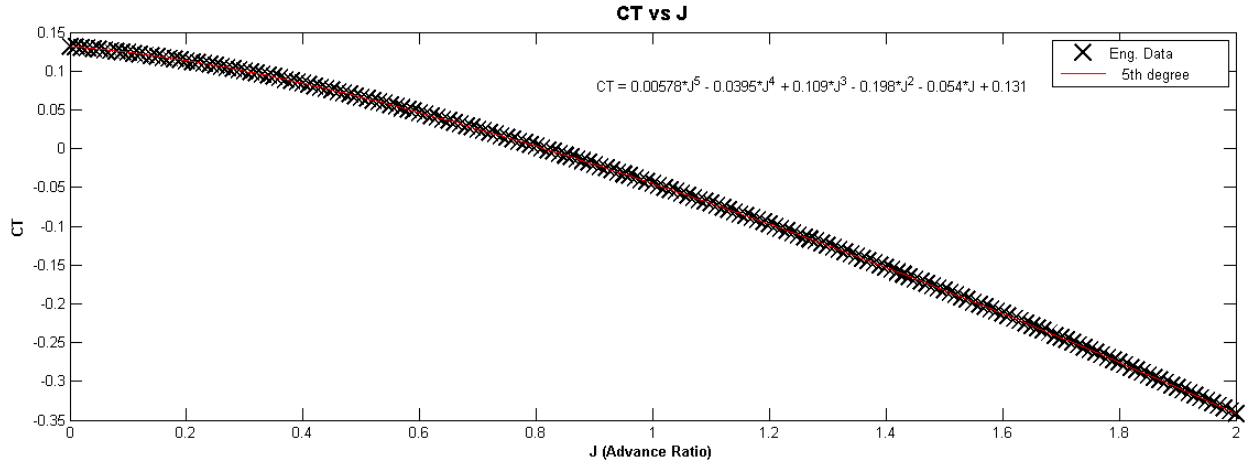

Figure 6.5: Curve Fitting to CT Data, experimental data (X), curve fit (red line)

A fifth order polynomial was fitted to the data points in figure 6.5 and the following expression was found:

$$\mathrm{CT(J)} = 0.00578\,\mathrm{J}^5 - 0.0395\,\mathrm{J}^4 + 0.109\,\mathrm{J}^3 - 0.198\,\mathrm{J}^2 - 0.054\,\mathrm{J} + 0.131. \tag{6.7}$$

Hence the approximation model of the aircraft for its longitudinal motion is:

body velocity components:

$$u = V_N \cos\theta - V_D \sin\theta\,, \tag{6.8}$$

$$w = V_N \sin\theta + V_D \cos\theta\,, \tag{6.9}$$

true airspeed:

$$V_T = \sqrt{{V_N}^2 + {V_D}^2}\,, \tag{6.10}$$

angle of attack:

$$\alpha = \arctan\left(\frac{w}{u}\right), \tag{6.11}$$

dynamic pressure:

$$\bar{q} = \frac{1}{2}\,\rho\,(V_T)^2\,, \tag{6.12}$$

aerodynamic forces and moments:

$$\begin{aligned}\mathrm{CM} = {} & 0.255\,\alpha^3 - 0.0638\,\alpha^2 - 0.603\,\alpha - 0.0305 \\ & + \mathrm{CM}_{\delta_E}\,\delta_E + \mathrm{CM}_q \left(\frac{c}{2\,V_T}\right) q,\end{aligned} \tag{6.13}$$

$$\begin{aligned}\mathrm{CL} = & -175\,\alpha^5 + 55.1\,\alpha^4 - 11.4\,\alpha^3 - 2.05\,\alpha^2 \\ & + 4.89\,\alpha + 0.533 + \mathrm{CL}_{\delta_E}\,\delta_E + \mathrm{CL}_q\left(\frac{c}{2\,V_T}\right) q,\end{aligned} \tag{6.14}$$

$$\begin{aligned}\mathrm{CD} = & \; 103\,\alpha^6 - 13.7\,\alpha^5 - 8.55\,\alpha^4 + 0.202\,\alpha^3 \\ & + 0.916\,\alpha^2 + 0.155\,\alpha + 0.0361,\end{aligned} \tag{6.15}$$

where $\mathrm{CL}_q = 7.8415$, $\mathrm{CL}_{\delta_E} = 0.3099$, $\mathrm{CM}_q = -14.3808$, $\mathrm{CM}_{\delta_E} = -0.8143$,

aerodynamic forces and moments in the body axis:

$$\mathrm{CX} = \mathrm{CL}\,\sin(\alpha) - \mathrm{CD}\,\cos(\alpha), \tag{6.16}$$

$$\mathrm{CZ} = -\mathrm{CL}\,\cos(\alpha) - \mathrm{CD}\,\sin(\alpha), \tag{6.17}$$

$$\mathrm{FX} = \bar{q}\,S\,\mathrm{CX}, \tag{6.18}$$

$$\mathrm{FZ} = \bar{q}\,S\,\mathrm{CZ}, \tag{6.19}$$

$$\mathrm{M}_p = \bar{q}\,S\,c\,\mathrm{CM}, \tag{6.20}$$

$$\tag{6.21}$$

engine forces and moments:

$$\omega = 857\,\delta_{\mathrm{th}}^4 - 1.45e+03\,\delta_{\mathrm{th}}^3 + 421\,\delta_{\mathrm{th}}^2 + 772\,\delta_{\mathrm{th}} - 2.94, \tag{6.22}$$

$$J = 2\,\pi\,\frac{V_T}{D\,\omega}, \tag{6.23}$$

$$\mathrm{CT} = 0.00578\,J^5 - 0.0395\,J^4 + 0.109\,J^3 - 0.198\,J^2 - 0.054\,J + 0.131, \tag{6.24}$$

$$\mathbf{F}_{\mathrm{eng}} = 2\left[CT\left(\frac{\omega}{2\pi}\right)^2 \rho\,D^4, \quad 0, \quad 0\right]^{\mathsf{T}}, \tag{6.25}$$

$$\mathbf{M}_{\mathrm{eng}} = \mathbf{r}_1 \times \frac{\mathbf{F}_{\mathrm{eng}}}{2} + \mathbf{r}_2 \times \frac{\mathbf{F}_{\mathrm{eng}}}{2}, \tag{6.26}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the position vectors of each engine from the c.g. Note that there are two engines, hence the multiplication of the engine force by a factor of 2.

Next we calculate the components of acceleration and the angular rates:

$$a_X = \frac{\mathrm{FX} + \mathrm{FX}_{\mathrm{eng}}}{m}, \tag{6.27}$$

$$a_Z = \frac{\mathrm{FZ}}{m}, \tag{6.28}$$

$$a_N = a_X \cos(\theta) + a_Z \sin(\theta), \tag{6.29}$$

$$a_D = -a_X \sin(\theta) + a_Z \cos(\theta) + g. \tag{6.30}$$

$$\tag{6.31}$$

Finally the state equations are:

$$\dot{x_D} = V_d, \tag{6.32}$$

$$\dot{V_N} = a_N, \tag{6.33}$$

$$\dot{V_D} = a_D, \tag{6.34}$$

$$\dot{\theta} = q \tag{6.35}$$

$$\dot{q} = \frac{\mathrm{M}_p + \mathrm{MY}_{\mathrm{eng}}}{I_Y}, \tag{6.36}$$

where I_Y is the moment of inertia with respect to the body y-axis and is equal to $18\,\mathrm{kg\ m^2}$. In the next section an analysis of the validity of the approximation model against the full order plant model is carried out.

6.3.2 Model Validation and Verification

Before the lower order approximation model could be used it was important to validate and verify the accuracy of the approximation model against the higher order plant model. For this purpose a proportional integral derivative (PID) controller was developed to control the plant model. The PID controller consists of a height control loop to calculate a demanded pitch angle given a desired height. The pitch demand is fed to the pitch control loop to calculate the required elevator deflection and the throttle input is obtained via a speed control loop.

The PID controller is used to control the plant model which is the full model comprising of the experimental data. For all validation tests the aircraft is required to fly the trajectory given in figure 6.6.

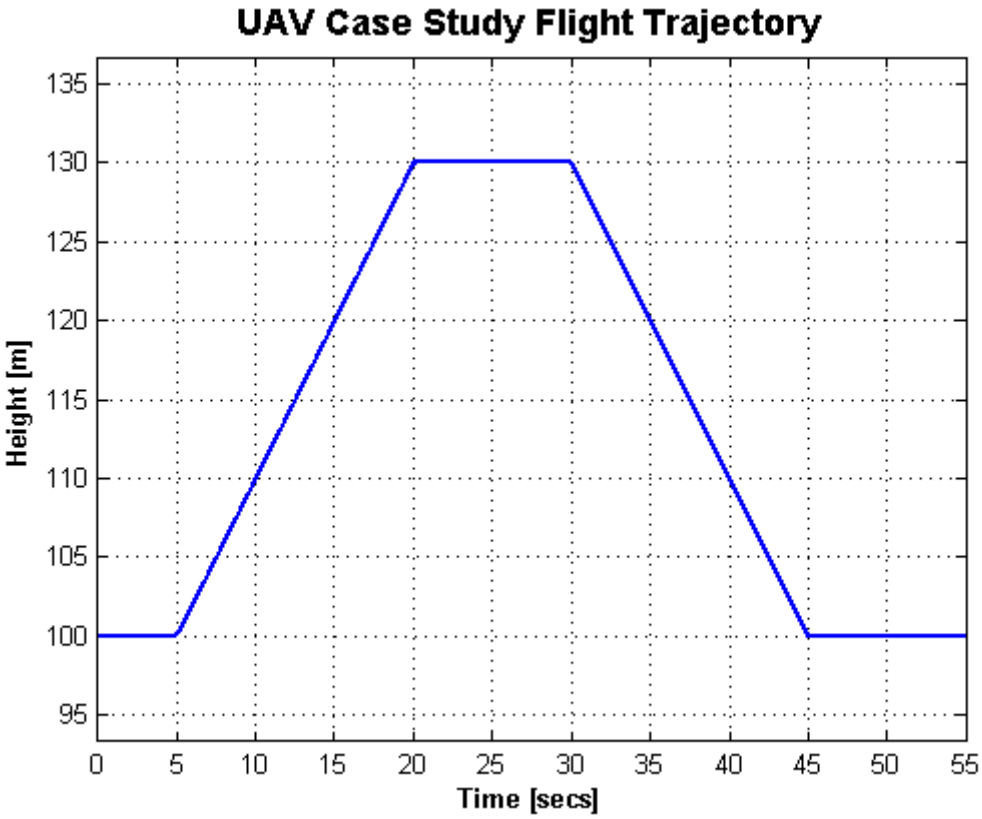

Figure 6.6: UAV Case Study Reference Trajectory

The first validation was to check if the algebraic expressions for the force and moment coefficients are acceptable. The control inputs produced by the PID controller to fly the plant model on the given trajectory (figure 6.6) were input into the approximation model and the force and moment coefficients from both models were compared. Figure 6.7 shows the CL values produced by the actual higher order model and the approximation model. The results show that the approximation model closely follows the response of the actual higher order model. Figures 6.8, 6.9, 6.10 and 6.11 show the comparison plots for CD, CM, RPM and CT respectively. All plots show that the approximation model does an excellent job of producing the same results. There are some differences which are to be expected, however they are within acceptable bounds.

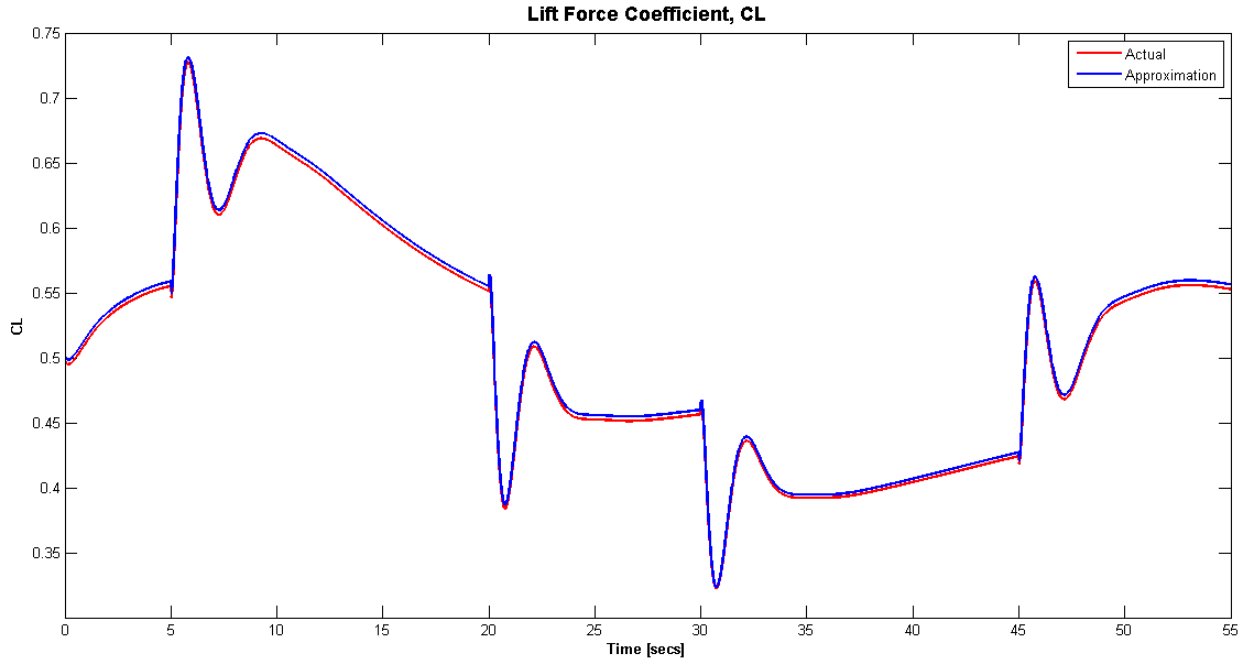

Figure 6.7: Comparison of CL, actual model (red), approximation model (blue)

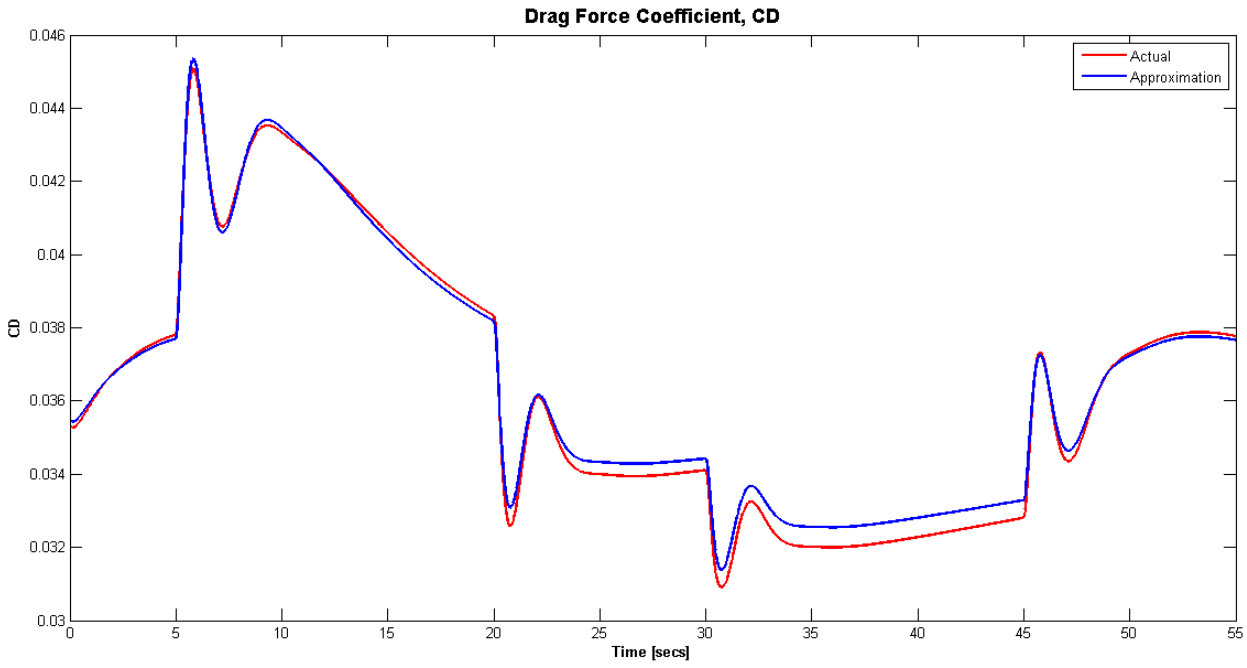

Figure 6.8: Comparison of CD, actual model (red), approximation model (blue)

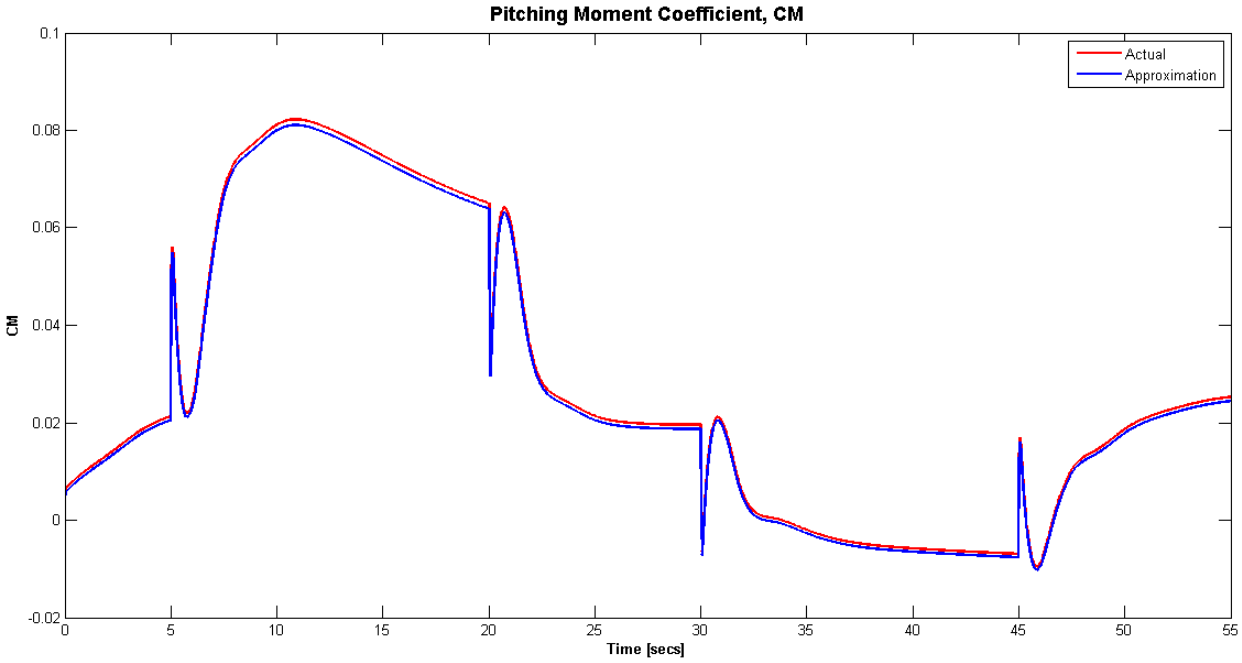

Figure 6.9: Comparison of CM, actual model (red), approximation model (blue)

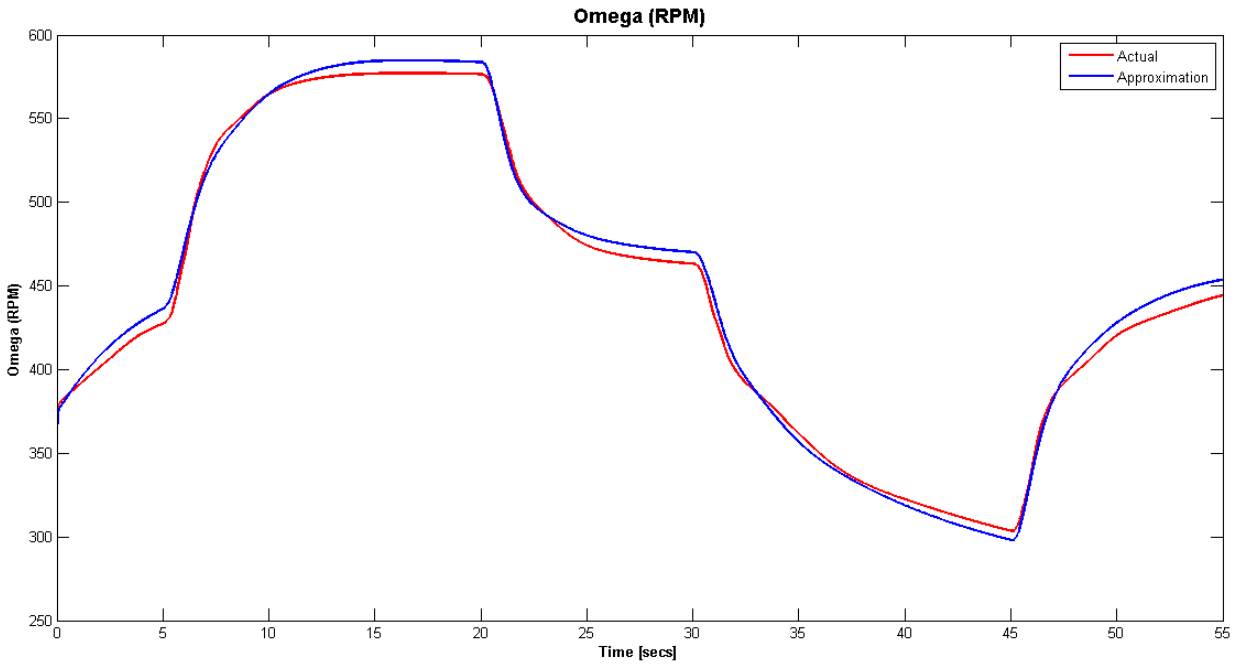

Figure 6.10: Comparison of RPM, actual model (red), approximation model (blue)

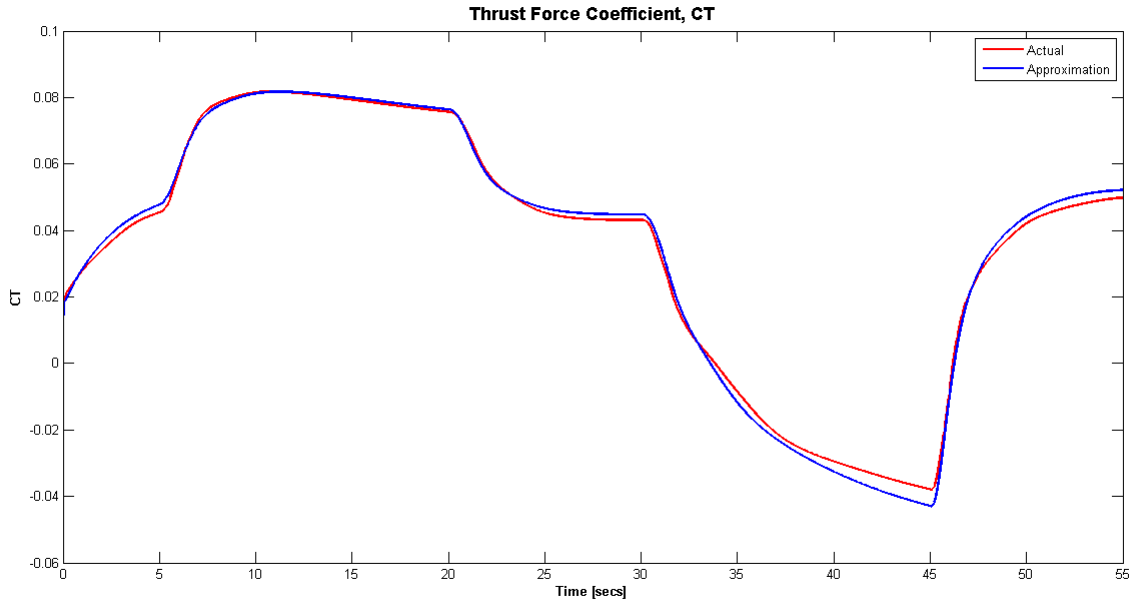

Figure 6.11: Comparison of CT, actual model (red), approximation model (blue)

The last step in verifying the validity of the approximation model is to confirm that if the approximation model were to be the plant model then similar control inputs are produced. The plots given in figures 6.12 and 6.13 show the comparisons of the throttle and elevator responses produced between the actual higher order model and the approximation model. Both figures show that the approximation is in compliance with the actual higher order model.

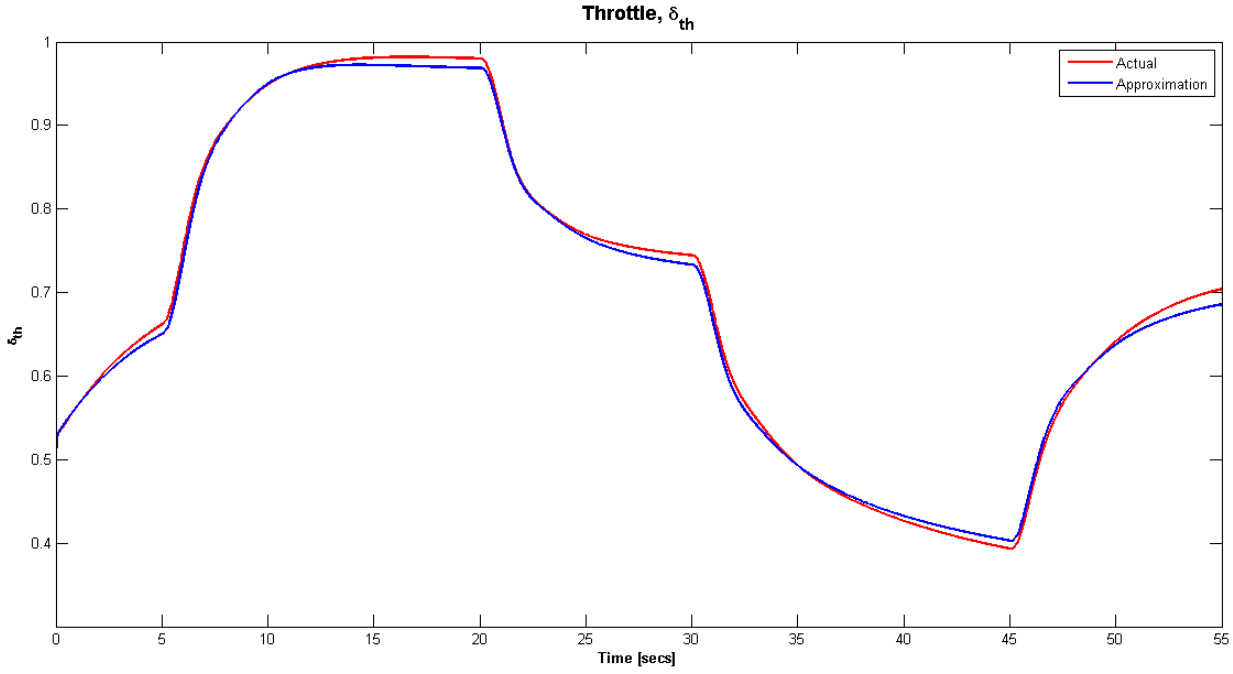

Figure 6.12: Comparison of Throttle Response, actual model (red), approximation model (blue)

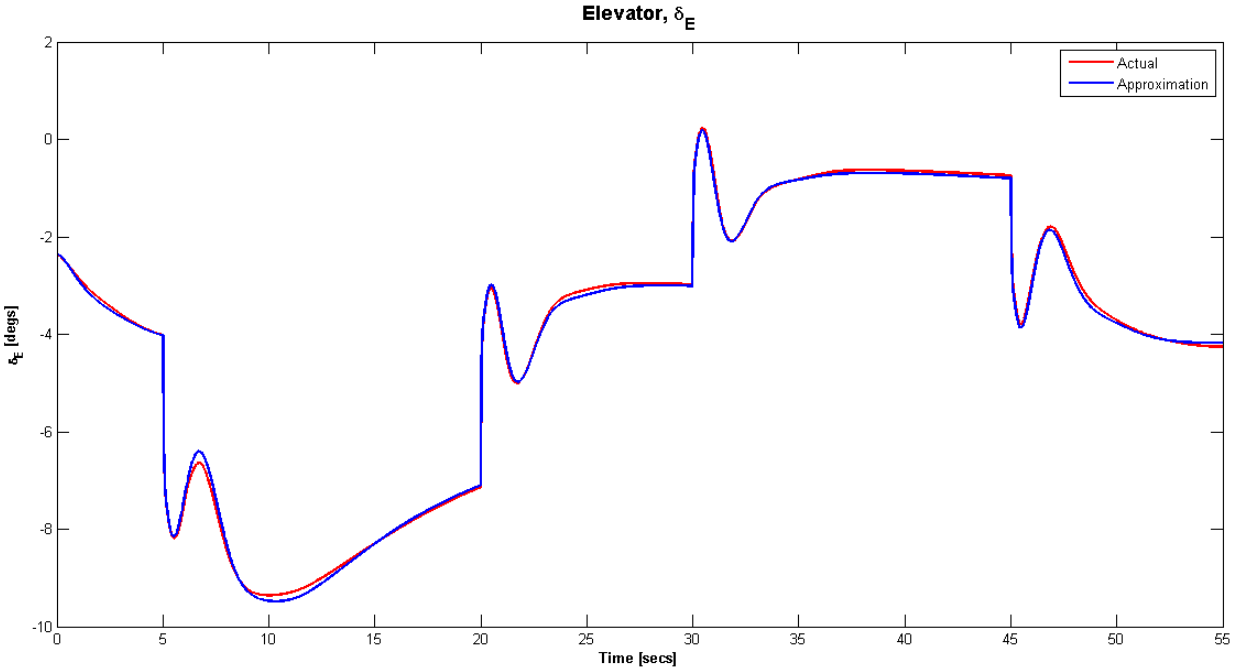

Figure 6.13: Comparison of Elevator Response, actual model (red), approximation model (blue)

The results produced in this section verify the validity of the approximation model. This means that the approximation can be used to predict the behaviour of the actual plant within the NMPC controller. The next section looks at the design of the NMPC controller.

6.3.3 NMPC Controller

Before applying the the active FTC system of section 5.7.2 a pseudospectral based NMPC controller is designed for the longitudinal motion of the UAV model using the current control inputs, δ_{th} and δ_E. The following optimal control problem is solved at each time step:

$$\min_{\mathbf{x},\mathbf{u}} \frac{H_p}{2} \sum_{j=1}^{j=N+1} \left(\left\|\mathbf{x}_D(j)-\mathbf{x}_{D_{\text{ref}}}(j)\right\|^2_{Q_x} + \left\|\mathbf{V}_T(j)-\mathbf{V}_{T_{\text{ref}}}(j)\right\|^2_{Q_{VT}} + \left\|\mathbf{V}_D(j)-\mathbf{V}_{D_{\text{ref}}}(j)\right\|^2_{Q_{VD}} + \left\|\Delta\mathbf{u}\right\|^2_{Q_{\Delta\mathbf{u}}} \right) w(j), \tag{6.37}$$

subject to

$$\left(\frac{t_f - t_0}{2}\right) \mathbf{D}_{j,k}\mathbf{x}(t) - \dot{\mathbf{x}} = 0, \tag{6.38}$$

$$\mathbf{x}(t_0) - \mathbf{x}_{\text{dem}}(t_0) = 0, \tag{6.39}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{6.40}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}, \tag{6.41}$$

$$\Delta\mathbf{u}_{\text{lb}} \leq \Delta\mathbf{u} \leq \Delta\mathbf{u}_{\text{ub}}, \tag{6.42}$$

where Q_x, Q_{VT}, Q_{VD} and $Q_{\Delta \mathbf{u}}$ are weighting parameters.

Table 6.2: Constraints for longitudinal Motion

Variable	**Upper Constraint**	**Lower Constraint**
x_D	300 m	1 m
V_N	26 m/s	15.6 m/s
V_D	3 m/s	−3 m/s
θ	None	None
q	None	None
δ_e	30 deg	−30 deg
δ_{th}	100%	0%
$\Delta\delta_{th}$	200 %/s	−200 %/s
$\Delta\delta_e$	60 deg/s	−60 deg/s

As initial conditions for the controller the trim conditions of the aircraft at a true airspeed of 20m/s are: $\theta = -0.0040$rads, $\delta_{th} = 0.7281$rads and $\delta_E = -0.0603$rads. To test the controller the aircraft was required to fly the reference trajectory given in figure 6.6. For the test cases the effects of wind have not been taken into account.

The control inputs produced by the NMPC controller are given in figure 6.14 and show that the inputs remain well within the constraints of the vehicle. The true airspeed and vertical speed (or climb rate) demands are given in figures 6.15 and 6.16 respectively. Both plots show that the controller does an excellent job at maintaining the velocity demands. This is further exemplified by the height profile given in figure 6.17 showing the aircraft successfully flying the demanded trajectory.

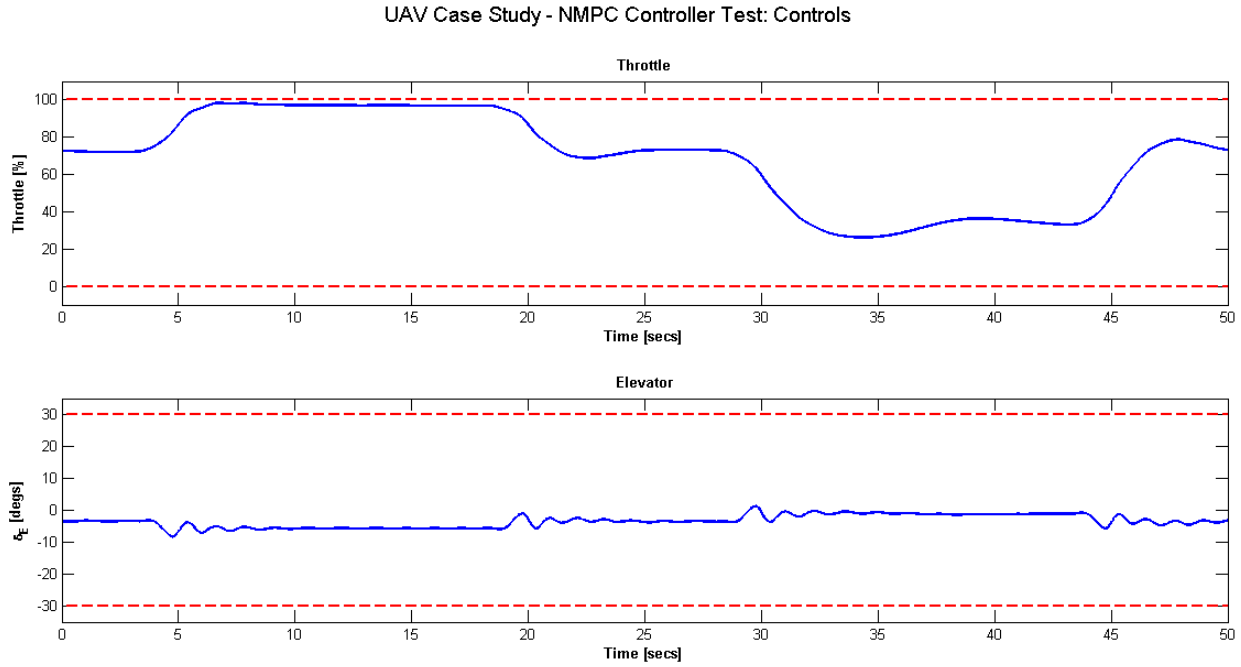

Figure 6.14: UAV Case Study NMPC Controller Test - Controls, constraints (dashed red lines), control inputs (blue)

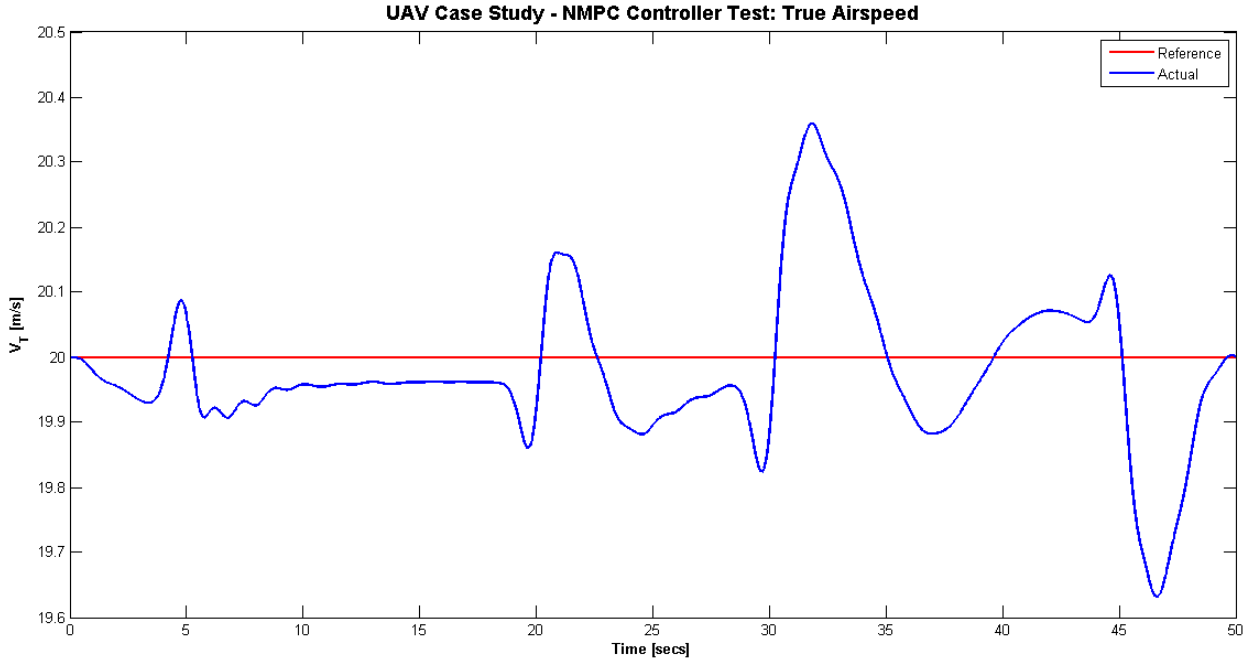

Figure 6.15: UAV Case Study NMPC Controller Test - True Airpseed, V_T

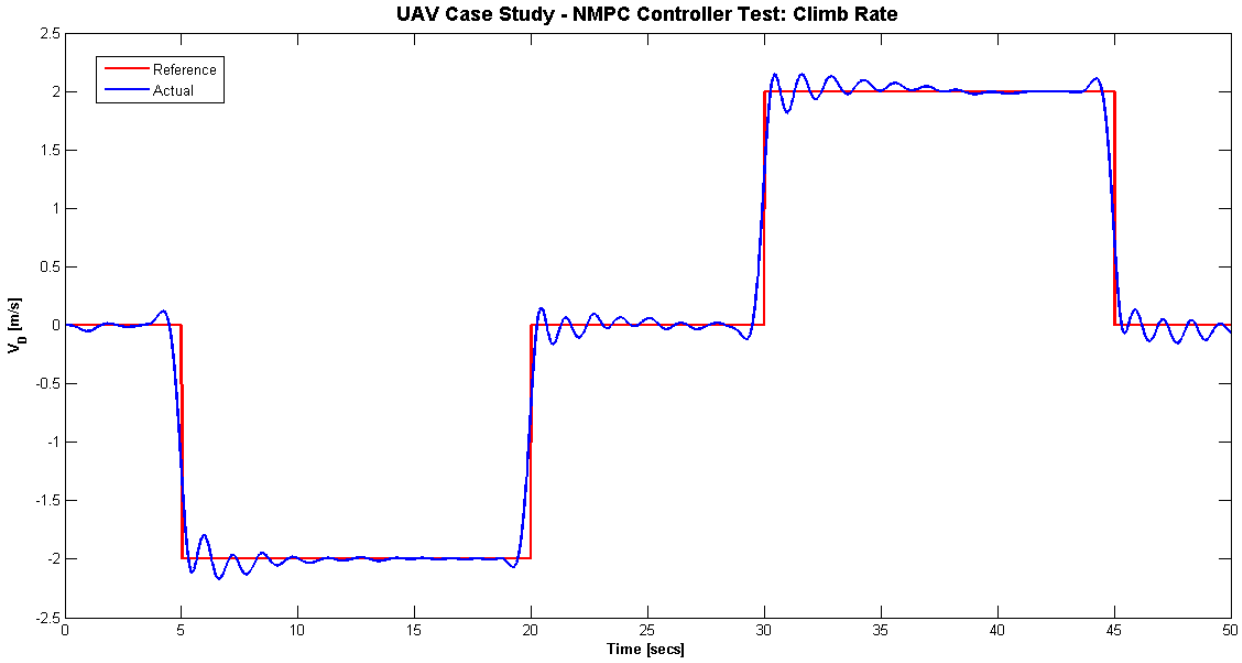

Figure 6.16: UAV Case Study NMPC Controller Test - Climb Rate, V_D

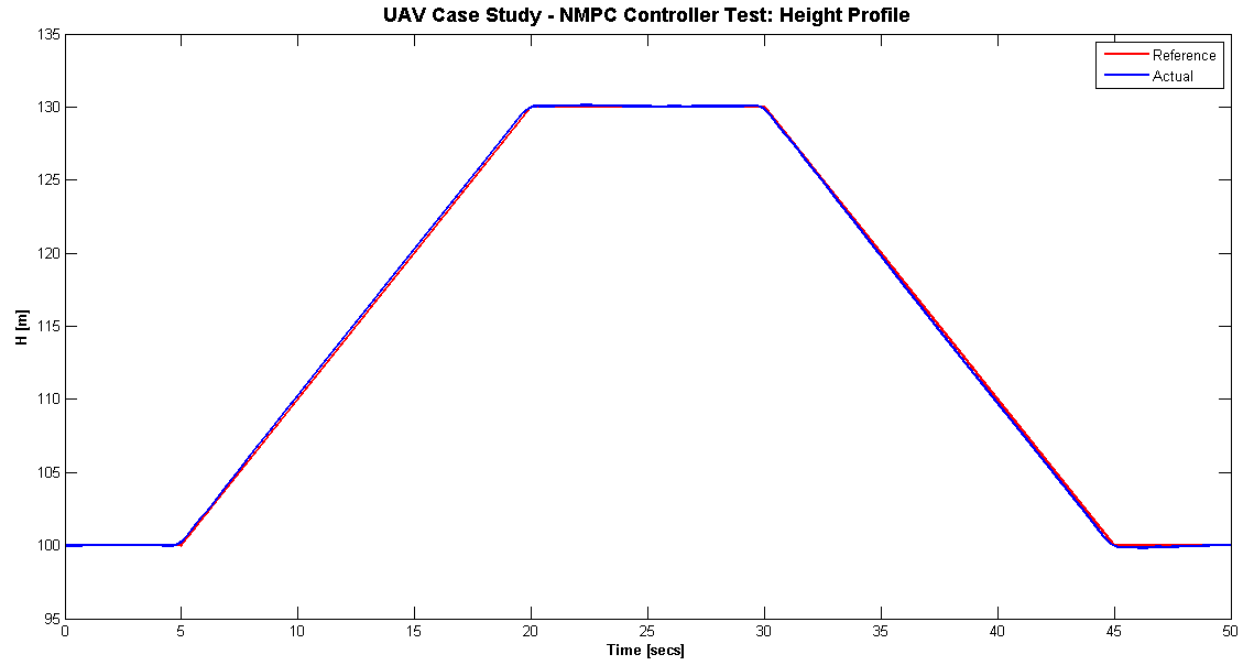

Figure 6.17: UAV Case Study NMPC Controller Test - Height Profile

The next section investigates the application of the active FTC system developed in section 5.7 to the given UAV model.

6.4 Application of Active FTC System Design to a UAV

The active FTC developed in 5.7 is based on a thrust NMPC controller. Hence the NMPC controller developed in the previous section is now converted to a thrust controller equivalent. Pseudospectral discretisation is applied to the controller design and the NMPC state vector is:

$$\mathbf{x_{nmpc}} = [x_D,\ V_N,\ V_D,\ \theta,\ q,\ \delta_{\text{thrust}},\ \delta_e,\ \Delta\delta_{\text{thrust}}\ \Delta\delta_e]^{\mathsf{T}}, \tag{6.43}$$

The following optimal control problem is solved at each time step:

$$\begin{aligned}\min_{\mathbf{x},\mathbf{u}} \frac{H_p}{2} \sum_{j=1}^{j=N+1} \Big(&\left\|\mathbf{x}_D(j) - \mathbf{x}_{D_{\text{ref}}}(j)\right\|^2_{Q_x} + \left\|\mathbf{V}_T(j) - \mathbf{V}_{T_{\text{ref}}}(j)\right\|^2_{Q_{VT}} \\ &+ \left\|\mathbf{V}_D(j) - \mathbf{V}_{D_{\text{ref}}}(j)\right\|^2_{Q_{VD}} + \left\|\Delta\delta_{\text{thrust}}\right\|^2_{Q_T} + \left\|\Delta\delta_e\right\|^2_{Q_{\delta_e}} \Big)\, w(j),\end{aligned} \tag{6.44}$$

subject to

$$\left(\frac{t_f - t_0}{2}\right) \mathbf{D}_{j,k}\mathbf{x}\,(t) - \dot{\mathbf{x}} = 0, \tag{6.45}$$

$$\mathbf{x}(t_0) - \mathbf{x}_{\text{dem}}(t_0) = 0, \tag{6.46}$$

$$\mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub}, \tag{6.47}$$

$$\mathbf{u}_{lb} \leq \mathbf{u} \leq \mathbf{u}_{ub}, \tag{6.48}$$

$$\Delta\mathbf{u}_{\text{lb}} \leq \Delta\mathbf{u} \leq \Delta\mathbf{u}_{\text{ub}}, \tag{6.49}$$

where V_T and $V_{T_{\text{ref}}}$ are the actual and reference true airspeeds respectively and Q_x, Q_{VT}, Q_{VD}, Q_T and Q_{δ_e} are weighting parameters and the terms $w(j)$ are the pseudospectral node weights as defined in chapter 3. The constraints applied are given in Table 6.3. The cost function weights are square matrices with the following diagonal values for each state: $Q_x = 10$, $Q_{VT} = 5$, $Q_{VD} = 5$, $Q_T = 0.01$ and $Q_{\delta_e} = 0.01$.

Table 6.3: Constraints for UAV Case Study, Thrust Controller

Variable	**Upper Constraint**	**Lower Constraint**
x_D	300 m	1 m
V_N	26 m/s	15.6 m/s
V_D	3 m/s	−3 m/s
θ	None	None
q	None	None
δ_e	30 deg	−30 deg
$\Delta\delta_{\text{thrust}}$	122 N/s	−122 N/s
$\Delta\delta_e$	60 deg/s	−60 deg/s

Trim conditions for the aircraft at 20m/s are: $\theta = -0.0040$rads, Thrust $= 27.7426\,N$ and $\delta_e = -0.0603$rads and have been used as initial conditions for the controller. The upper limit on the thrust constraint continually changes based on the true airspeed of the aircraft according to the equations given in (6.22). The minimum thrust level is at all time set to $0N$. If however an engine failure is detected the upper limit on thrust is set to the filter estimate plus 2σ uncertainty.

An unscented Kalman filter (UKF) filter was designed to perform fault detection and identification (FDI) with the following process noise and noise covariance matrices:

$$Q = \begin{bmatrix} (2\Delta t)^2 & 0 & 0 & 0 \\ 0 & (2\Delta t)^2 & 0 & 0 \\ 0 & 0 & (0.017\Delta\, t)^2 & 0 \\ 0 & 0 & 0 & (122\Delta t)^2 \end{bmatrix}, \qquad R = \begin{bmatrix} (0.5)^2 & 0 & 0 \\ 0 & (0.5)^2 & 0 \\ 0 & 0 & (0.17)^2 \end{bmatrix}, \quad (6.50)$$

where Δt is the filter update rate 0.01 secs. The initial state vector and covariance matrix are:

$$\mathbf{x}(0) = [20,\ 0\ -0.0040,\ 27.7426]^\intercal, \quad \mathbf{P}(0) = \begin{bmatrix} (0.5)^2 & 0 & 0 & 0 \\ 0 & (0.5)^2 & 0 & 0 \\ 0 & 0 & (0.0850)^2 & 0 \\ 0 & 0 & 0 & (6)^2 \end{bmatrix}. \quad (6.51)$$

Finally the fault detection logic is based on that given in subsection 5.7.1.4.

6.4.1 Numerical Results

The following scenarios were set up to test the active FTC system on the UAV model:

Scenario 1: no fault case

Scenario 2: engine failure - 50% power loss 20 seconds into flight,

Scenario 3: engine failure - 70% power loss 30 seconds into flight.

The aircraft was required to follow the flight trajectory given in figure 6.18 (Note: wind effects have been taken into account).

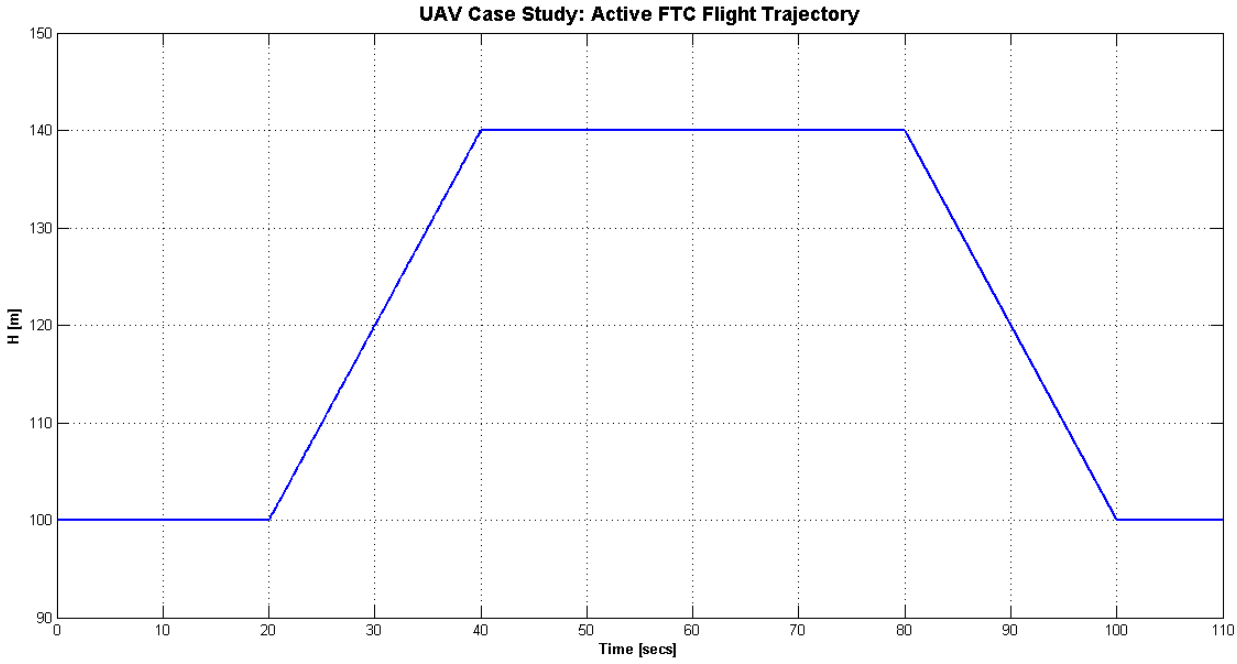

Figure 6.18: UAV Case Study Active FTC: Reference Trajectory

Figure 6.19 presents the fault detection logic for each scenario. There was no fault present in scenario 1 and this is reflected in the results of the fault detection logic as the value of the fault flag remained zero throughout the duration of the flight. The results for scenarios 2 and 3 show that the fault flag is triggered (ie. the value of the flag switches to 1) within seconds of the fault occurring.

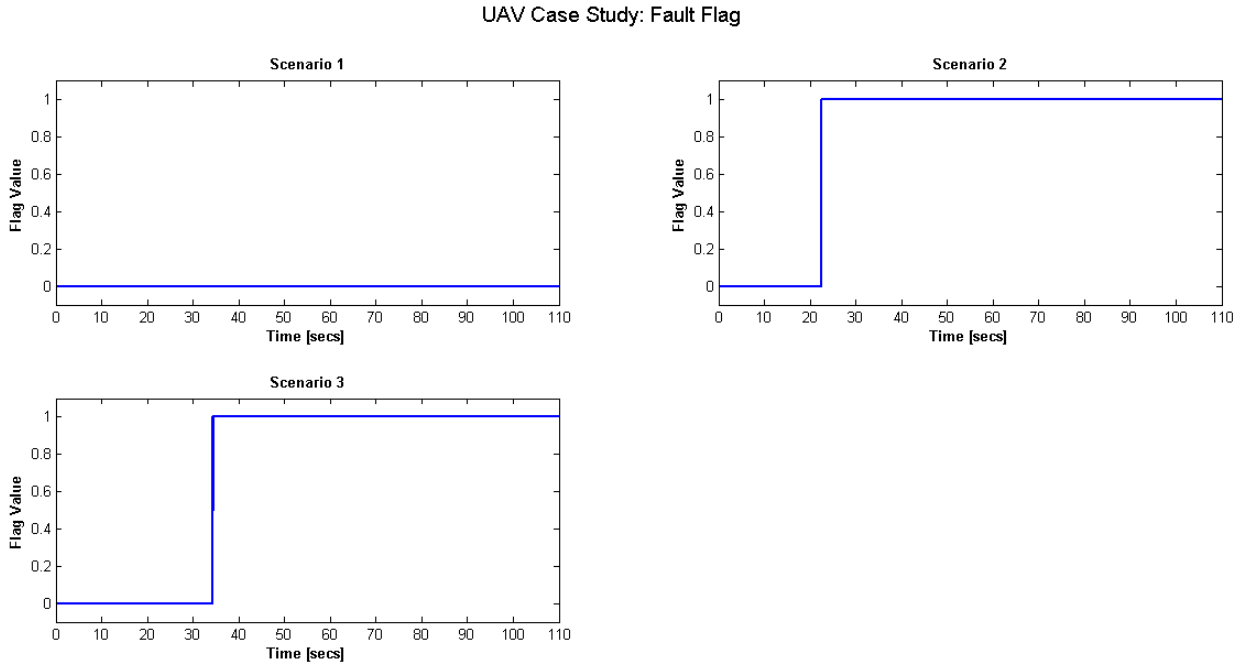

Figure 6.19: UAV Case Study Active FTC: Fault Detection Logic Results

The control inputs produced by the controller for each scenario are given in figures 6.20, 6.21 and 6.22. The results clearly indicate that the controller is able to successfully reconfigure based on FDI data and as the loss of power increases the demand on the elevator increases.

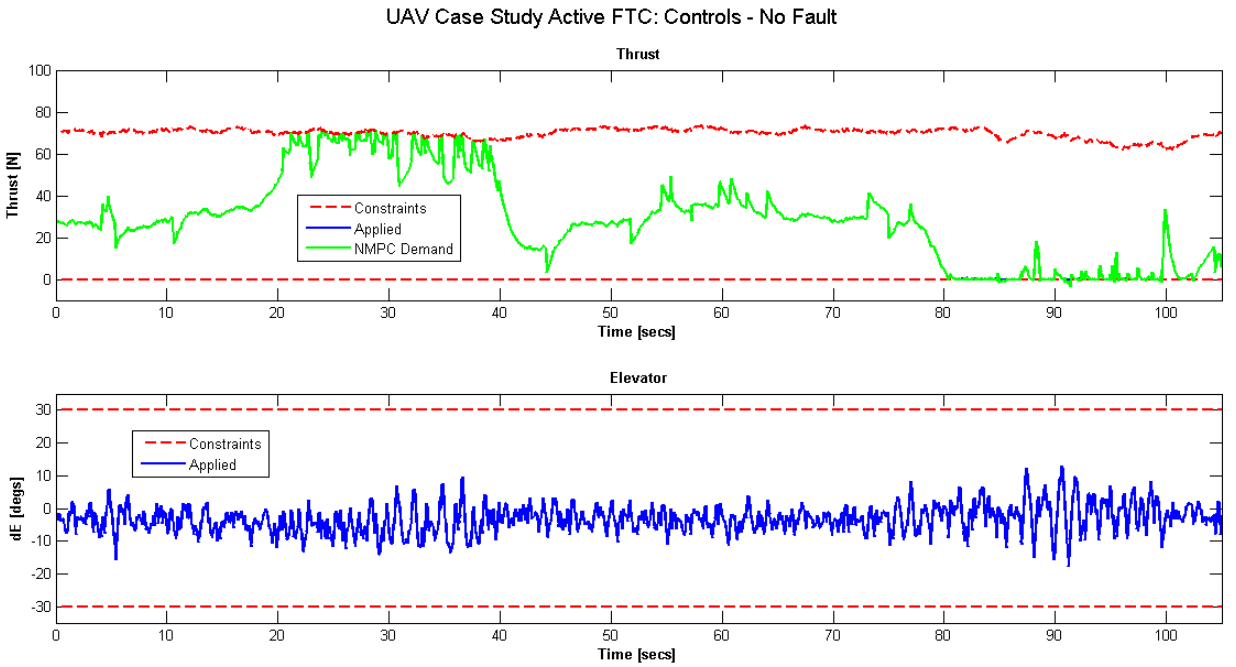

Figure 6.20: UAV Case Study Active FTC – Scenario 1: No Fault - Controls

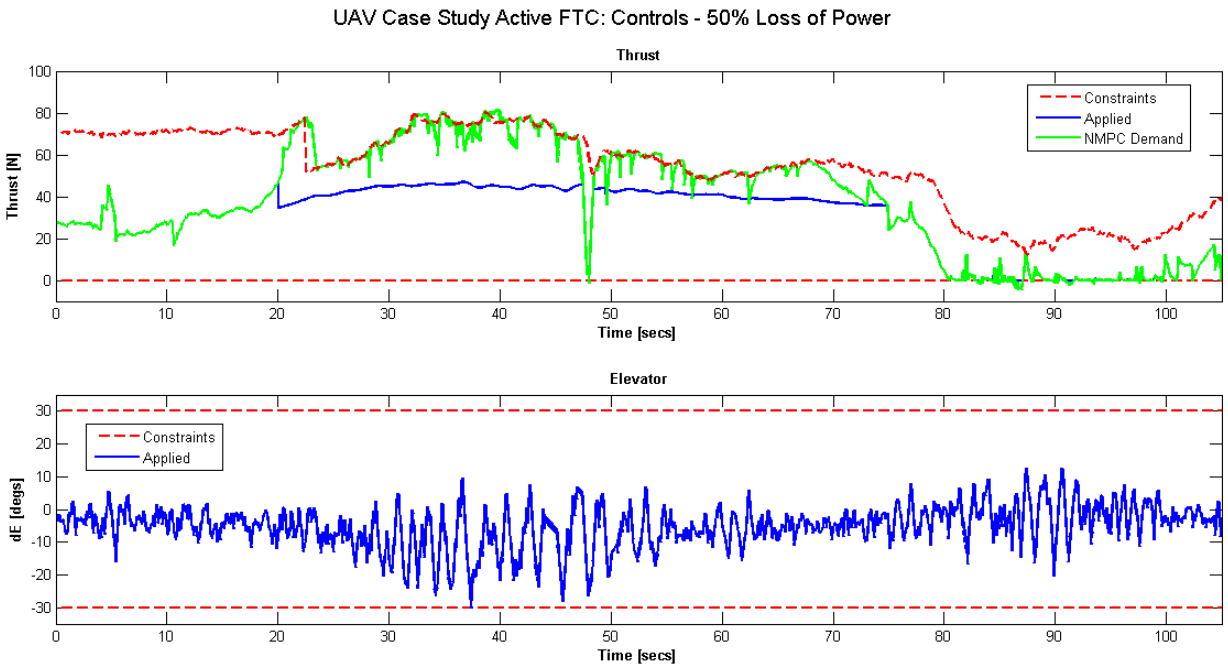

Figure 6.21: UAV Case Study Active FTC – Scenario 2: 50% Loss of Power - Controls

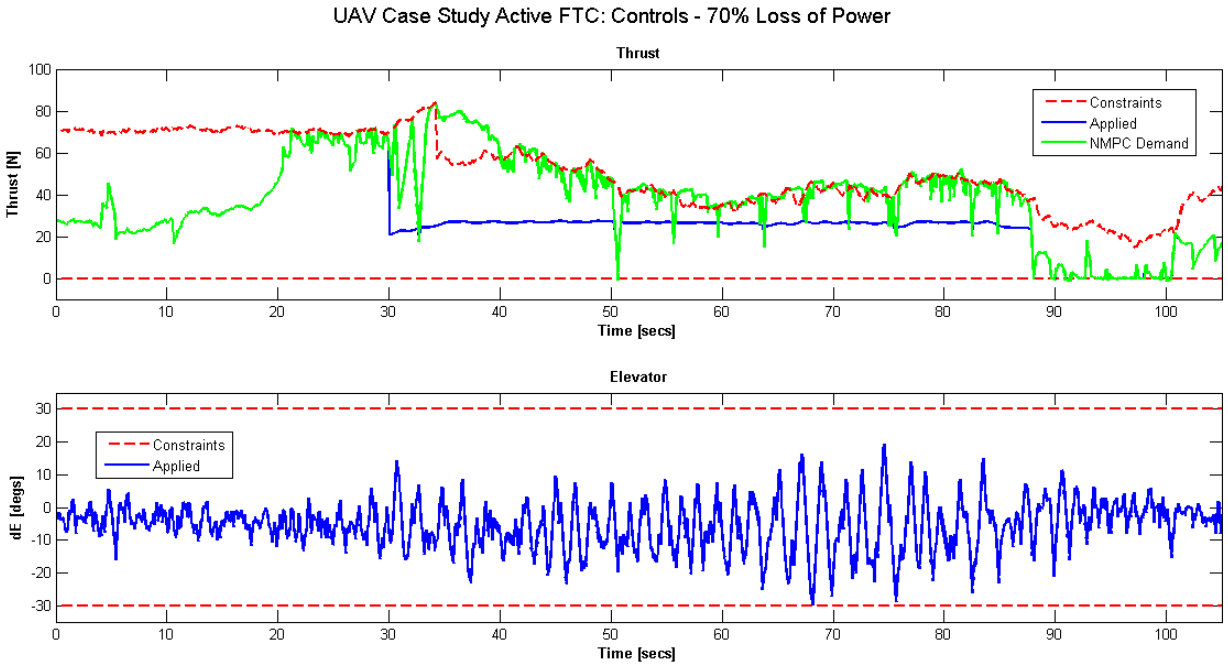

Figure 6.22: UAV Case Study Active FTC – Scenario 3: 70% Loss of Power - Controls

The true airspeed response of the aircraft is given in figure 6.23. The demanded speed was $20m/s$ and the results show that for a 50% loss in power the true airspeed demand is unachievable during climb to altitude. However during straight and level flight the velocity demand is gradually

achieved and is finally reached once the aircraft begins the descent phase. During the descent the true airspeed demand is similar to the no fault case. In the case of 70% power loss the aircraft is unable to meet the true airspeed demand during straight and level flight, however halfway through the descent phase the aircraft picks up speed and is able to maintain the reference. The main point to note here is that the aircraft stall speed of $12m/s$ was never reached hence the controller was able to avoid a stall situation even with 70% power loss.

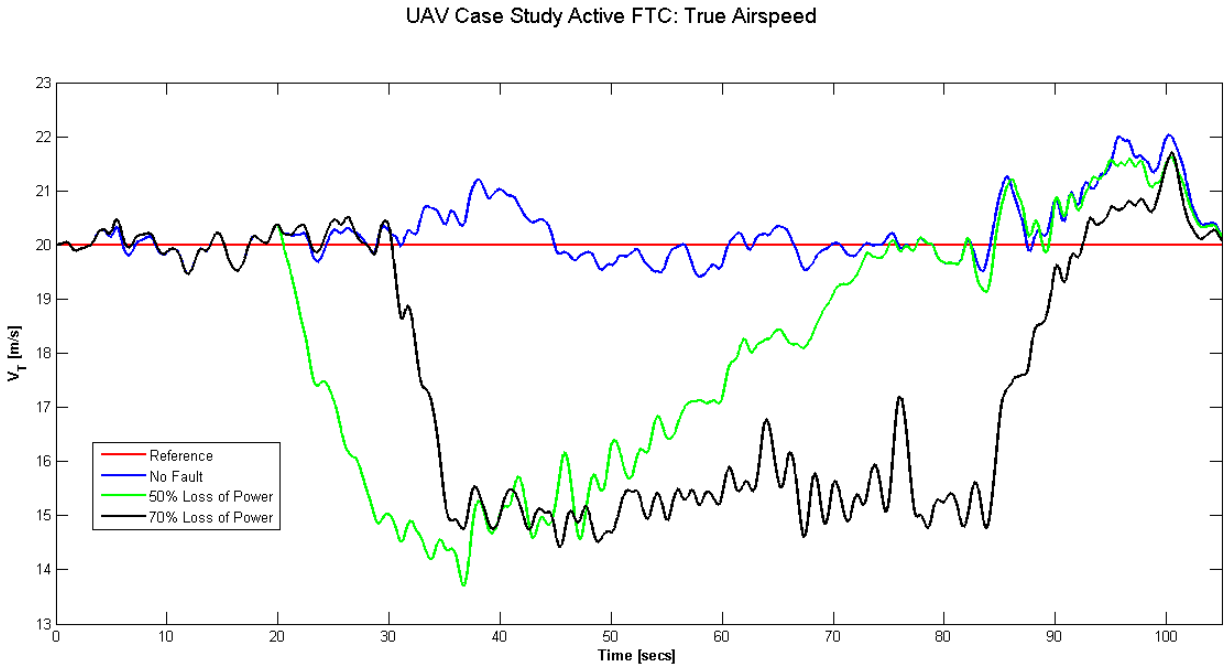

Figure 6.23: UAV Case Study Active FTC: True Airspeed, V_T

The climb rate (or vertical speed) response is shown in figures 6.24 and 6.25 and shows that during power loss the vertical speed oscillates between the upper and lower constraints values. This is to be expected as the elevator is working harder to regulate the speed, and hence bounces between the two limits.

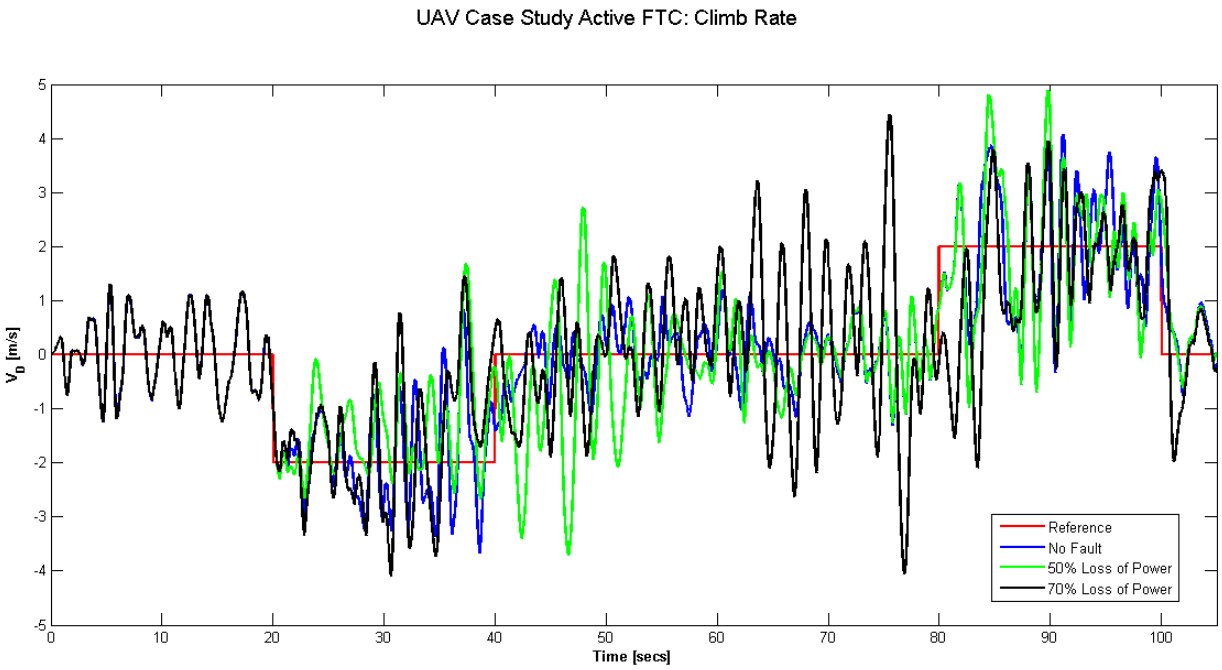

Figure 6.24: UAV Case Study Active FTC: Climb Rate, V_D

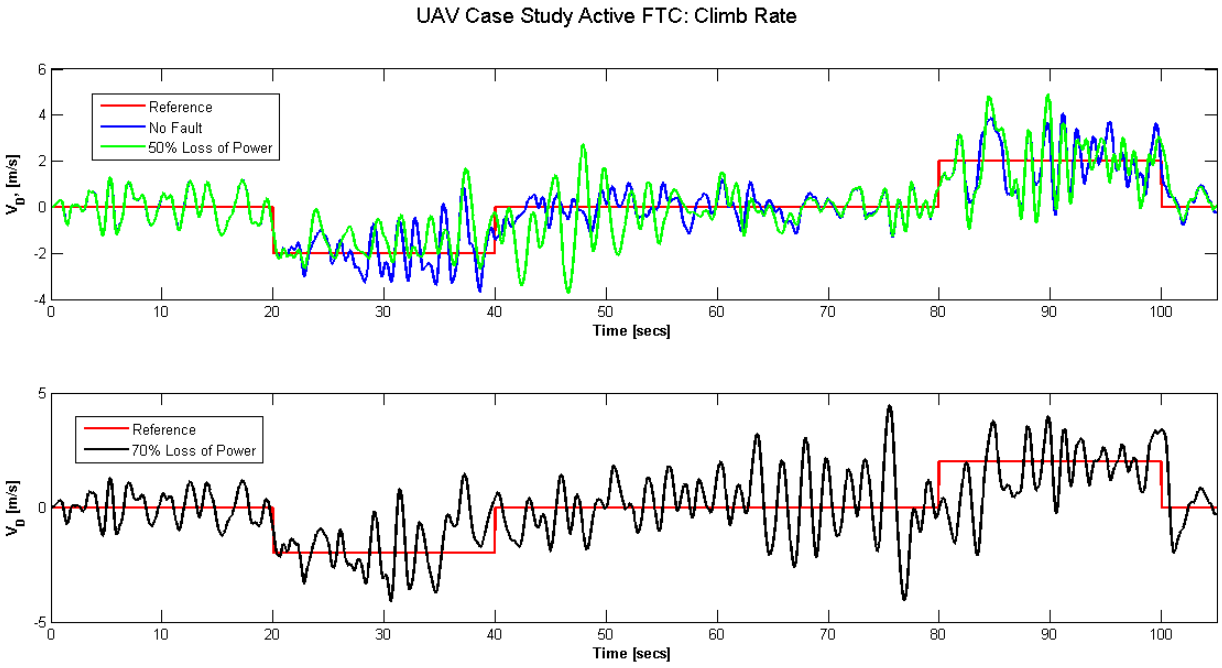

Figure 6.25: UAV Case Study Active FTC: Climb Rate, V_D

Plots for pitch rate, pitch angle, Angle of Attack (AoA) and g-forces (a_Z) have been provided in figures 6.26, 6.27, 6.28 and 6.29 respectively. There are no design constraints placed on these parameters however the plots show that even in the extreme failure case (70% power loss) the

aircraft maintains a reasonable pitch rate, θ and AoA. Also in the worst case scenario the g-forces experienced is 2g's for a very short period of time which is very unlikely to lead to structural damage.

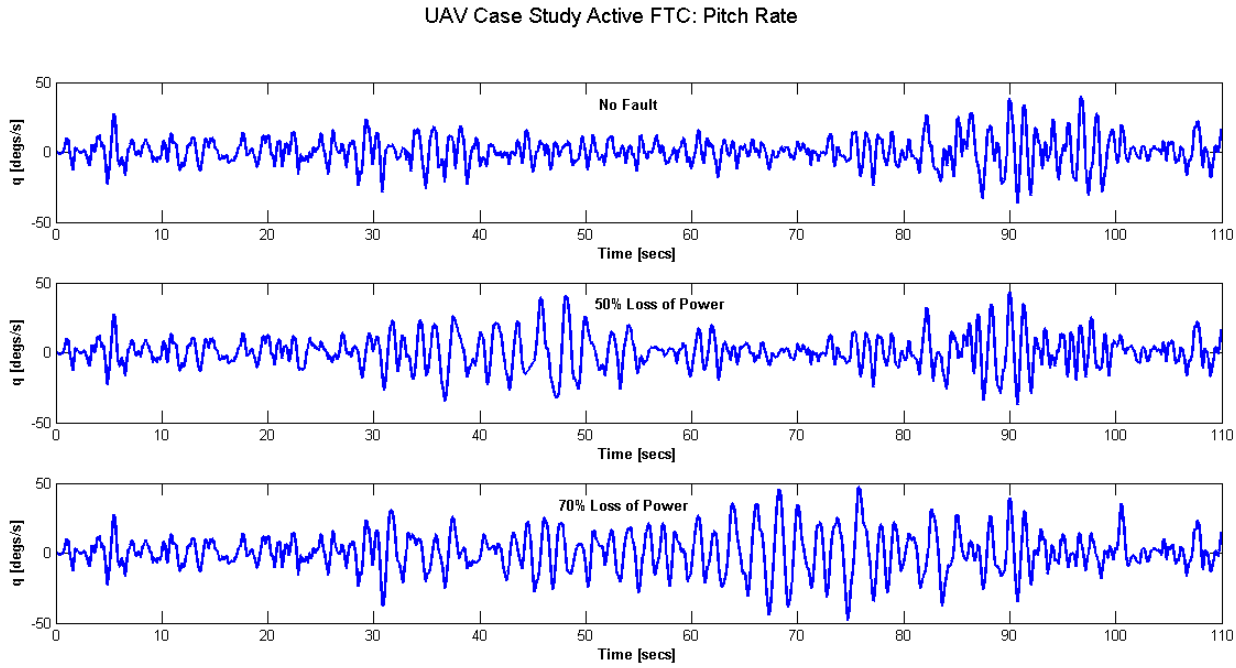

Figure 6.26: UAV Case Study Active FTC: Pitch Rate, q

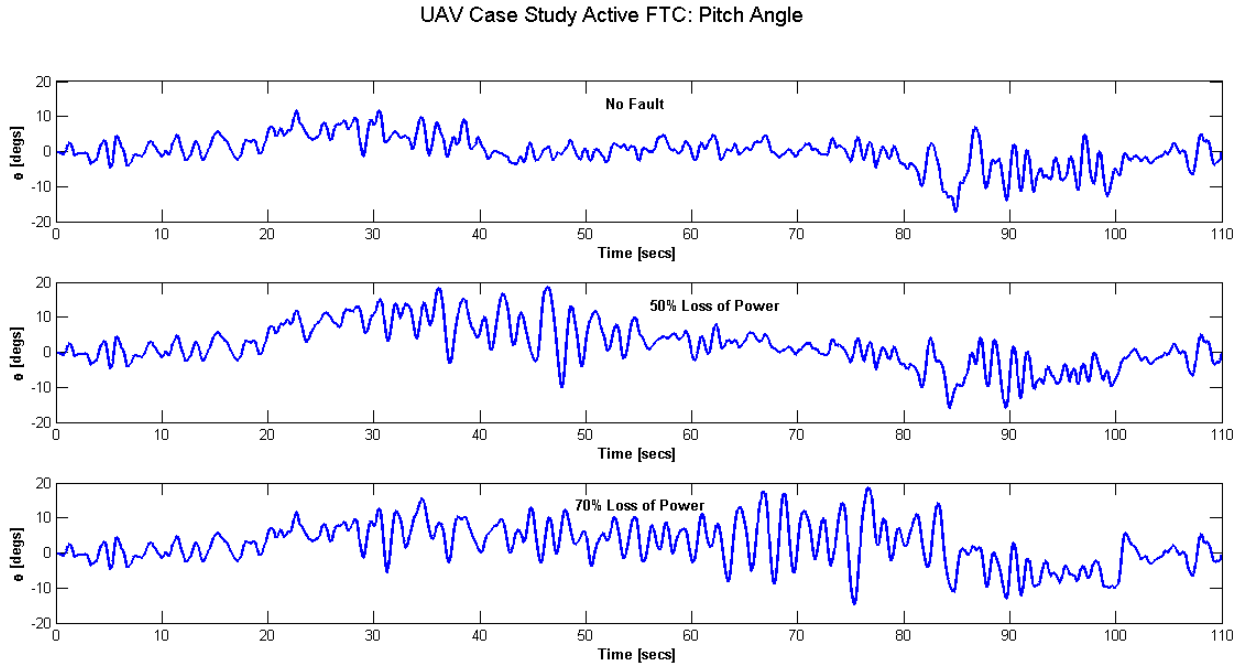

Figure 6.27: UAV Case Study Active FTC: Pitch Angle, θ

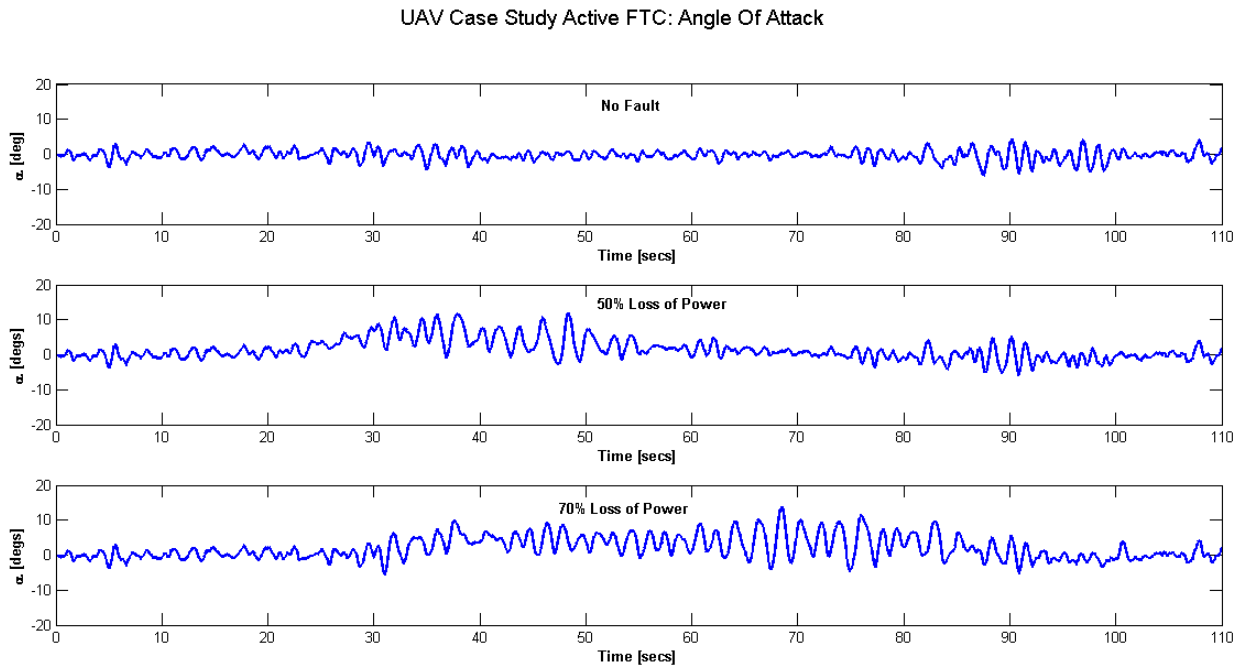

Figure 6.28: UAV Case Study Active FTC: Angle of Attack, α

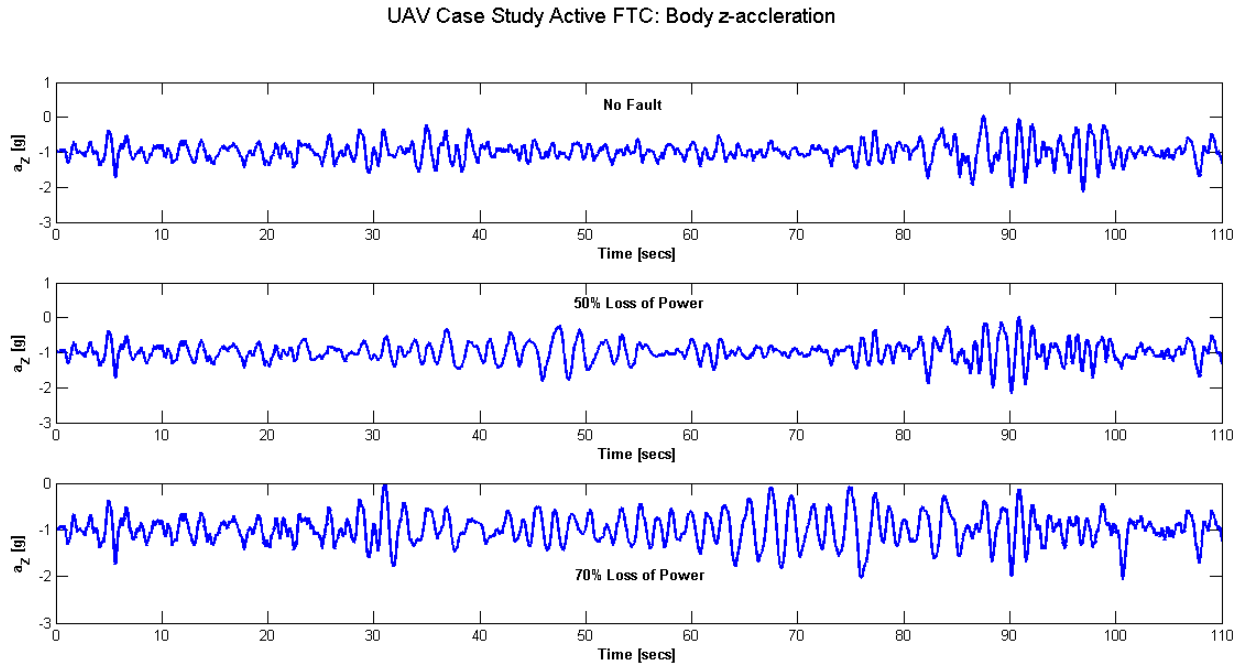

Figure 6.29: UAV Case Study Active FTC: Z-body Acceleration, a_Z

Finally the trajectories flown by the aircraft in each sceanrio are shown in figrure 6.30. The interesting points to note here are that in the highest loss of power case (scenario 3) the aircraft did not have enough power to reach the highest flight altitude so instead cruised at an altitude it

was capable of flying. Once the straight and level phase of the flight was over the controller was able to fly the aircraft back onto the demanded trajectory. The 50% loss of power case shows that the aircraft still had enough power to fly back onto the demanded path halfway through the straight and level flight phase.

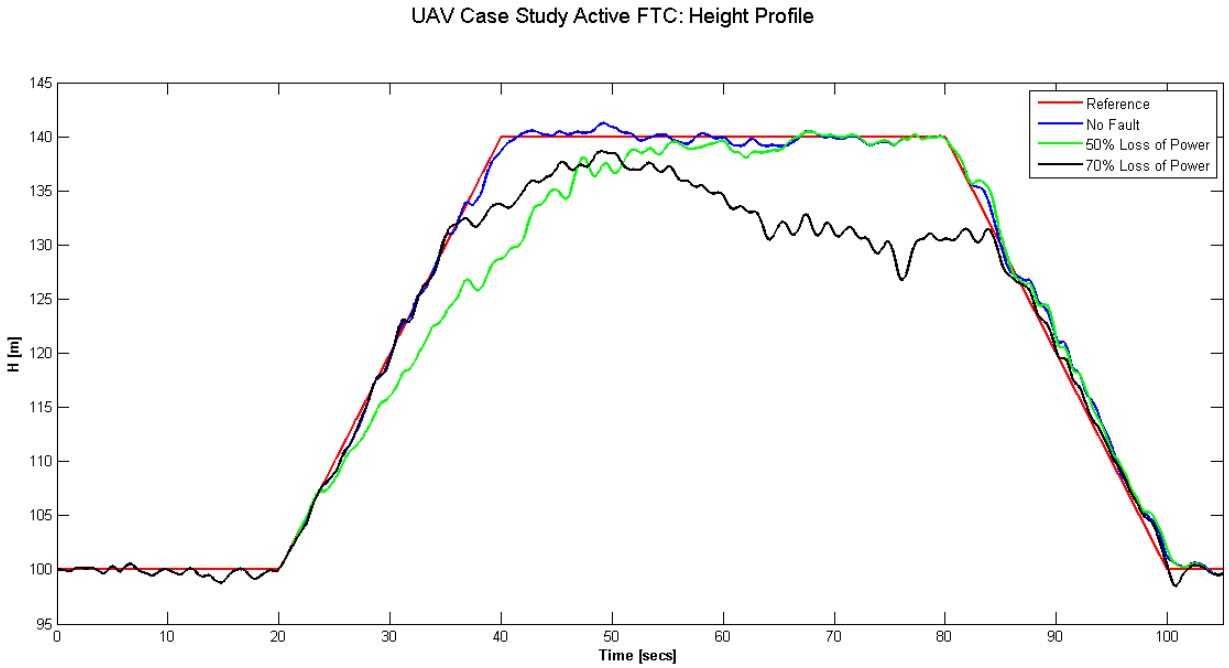

Figure 6.30: UAV Case Study Active FTC: Height Profile

6.5 Summary of Findings

This chapter demonstrated the successful application of my active FTC design on an actual UAV which is currently in operation. The controller was implemented using actual aircraft data and shows great promise for active fault tolerant flight control.

Chapter 7

Conclusion and Recommendations for Further Work

7.1 Conclusions

For many years linear model predictive control (MPC) has been the focus of research but very little attention has been paid to nonlinear MPC due to its greater computational load. However, due to the availability of faster computing and the benefits of incorporating a nonlinear prediction model in the controller design, this research focused on the implementation of nonlinear model predictive control (NMPC). More specifically, this thesis looked at the application of fault tolerant control (FTC) to flight control systems.

Receding horizon control (MPC) whether linear or nonlinear involves solving an open loop optimal control problem at every time step and various numerical techniques used for implementing optimal control were investigated in the design of the NMPC controller. The various designs were applied to the well known Brachistochrone problem and the collocation method using pseudospectral discretisation was found to be the most promising solution for fault tolerant NMPC design. This conclusion was mainly based on the fact that the pseudospectral technique produced accurate results with fewer discretisation points, thereby using less CPU time. Need for excessive CPU time is a major drawback of NMPC hence any saving in computational time is highly advantageous. The pseudospectral method was then used to design a linear as well as a nonlinear MPC controller for a simple 2D robot model using 50 discretisation points. The controllers were initially used to solve an open loop problem and the effects of selecting a cost function, integration time step, prediction window length and initial conditions were all analysed

and comparisons were made between the linear and nonlinear solutions. Based on the findings a prediction window length of 5 seconds was chosen as suitable for trajectory following. This was followed by an investigation into solving the closed loop problem and once again the linear and nonlinear outcomes were compared. All results showed that the linear solution performed on par with the nonlinear solution when the perturbations were small. However when perturbations in the linear controller became large the linear solution diverged. The design of the nonlinear controller proved to be the most promising solution particularly in the event when the linear assumptions, on which the linear controller is based, break down (ie. for large perturbations).

The next step in designing a fault tolerant controller was the investigation of fault detection techniques. Observation based techniques were implemented, specifically the extended Kalman filter (EKF), unscented Kalman filter (UKF) and the interactive multiple model (IMM) filters. The NMPC controller design was integrated with each of the filters and the performance of the overall active fault tolerant control system design was examined. The IMM filters outperformed the single EKF and UKF designs, however the single UKF filter was chosen for the final design, as although the IMM showed great potential as a fault detection scheme, designing a filter for all possible fault scenarios was deemed to be impractical. The final design of the fault tolerant controller using pseudospectral NMPC with a UKF filter for fault detection and identification (FDI) was compared to its linear counterpart where a linear MPC controller was integrated with a UKF filter. The results showed that in the event of a fault the performance of the linear filter was the same as the nonlinear solution up until the point at which the linear assumptions started to breakdown. The proposed nonlinear MPC design integrated with a UKF filter turned out to be the most viable solution to the active fault tolerant control problem.

The final design was then implemented on a generic aircraft model. Initially only the longitudinal motion of the aircraft was investigated assuming FDI information was available. Once the controller was tested and verified it was then applied to the full 6DoF aircraft where once again the FDI information was assumed. Once it was established that the controller was performing well as a fault tolerant controller the next step was to integrate the UKF FDI filter design. The filter was designed separately to the NMPC controller using the well known proportional integral derivative (PID) control methodology. The filter was designed to make estimates of the aircraft control derivatives which are directly affected in the event of a fault. The results showed that estimating these derivatives is not a simple task and goes beyond the scope of this

research. To remain within the scope and to demonstrate the feasibility of my proposed solution as fault tolerant flight controller, a full active FTC system was designed and implemented for the longitudinal motion of the aircraft. The scenario in which the engine fails and power to the aircraft is lost, was set up. The NMPC controller was designed to control thrust and elevator input and the UKF filter was tailored to estimate thrust level. If the thrust demand from the NMPC controller is greater than the filter thrust estimates for a specified period of time this indicates a fault and raises a fault flag. Once the flag has been set the thrust constraints of the controller are amended to the filter estimates, providing the controller with up to date information on the current status of the aircraft to allow it to optimally allocate control authority. The results show that my proposed design for active FTC flight control is viable.

The final phase of demonstrating design feasibility was to investigate the performance of the FTC system on an actual unmanned aerial vehicle (UAV) model. The aircraft model was provided by Williams [128] and is of an actual UAV currently in operation. The proposed active FTC design was successfully applied to the longitudinal motion of the aircraft. Results showed that the proposed design is a feasible methodology for the solution of fault tolerant flight control design for UAVs.

In conclusion, with regards to the research questions outlined in Chapter 2, nonlinear model predictive control was found to be a feasible solution to the fault tolerant flight control problem. Comparisons with linear MPC showed that in the event of a fault occurring for which nonlinearities in the system significantly affect the system behaviour, nonlinear MPC outperforms its linear counterpart. The NMPC solution was seen to have inherent fault tolerant capabilities which was demonstrated by the 6DoF model, where, in the event of an actuator failure and no FDI information, it was able to redistribute control authority, however with FDI information reaction time was found to be much shorter. It is common practice to design a nominal controller for fault free operations and a controller to handle faults which comes into effect once the FDI system detects a fault. Due to the inherent fault tolerant control capabilities of MPC it was found that there was no need to design two types of controllers. Thus, the inherent fault handling characteristics make MPC robust to faults up to a certain degree. If however fault information is available it can be used to reconfigure the controller. Overall the proposed active fault tolerant flight control system design shows promise as a practical method for fault tolerant flight control for UAVs.

7.2 Recommendations

As a result of the research in this thesis the following areas have been identified as having great merit for further research and development.

- Investigation of FDI for 6DoF aircraft motion and addressing the observability issue present in the 6DoF analysis.
- Investigation of other nonlinear fault detection techniques such as particle filtering and neural networks.
- Development of a guidance system where information from the FTC system can be incorporated into mission replanning. This is very important, as based on the health of the aircraft, the guidance system can decide if the aircraft is capable of flying the full mission or whether it is better to re-plan.
- Investigation of the handling of incorrect fault detection data such as false or delayed information.

Fault tolerant flight control is a difficult problem. Integration of both the controller and FDI components into the full 6DoF aircraft model is highly complex and requires a lot of time, multiple design procedures and travelling down many different paths to reach a full solution.

Bibliography

[1] IEE two-day workshop on model predictive control: Techniques and applications - day 1 (ref. no. 1999/095). `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?reload=true&punumber=6258`, note = Last Accessed: 07-01-2016, organization = IEE, 1999.

[2] Chae Ik Ahn, Youdan Kim, and Hyoun Jin Kim. Adaptive sliding mode controller design for fault tolerant flight control system. In *Proceedings of the 2006 AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 1–8, 2006.

[3] Muhammad Ahsan and Sarah Farrukh. A new type of shooting method for nonlinear boundary value problems. *Alexandria Engineering Journal*, 52(4):801 – 805, 2013.

[4] Frank Allgöwer and Alex Zheng. *Nonlinear Model Predictive Control*, volume 26 of *Progress in Systems and Control Theory*. Birkhuser Basel, 2000.

[5] H. Alwi, C. Edwards, O. Stroosma, and J.A. Mulder. SIMONA flight simulator implementation of a fault tolerant sliding mode scheme with on-line control allocation. In *Proceedings of the 2008 American Control Conference*, pages 1594–1599. IEEE, 2008.

[6] H. Alwi, C. Edwards, O. Stroosma, and J.A. Mulder. A simulator evaluation of a model-reference sliding mode fault tolerant controller. In *Control and Automation, 2009. ICCA 2009. IEEE International Conference on*, pages 878–883. IEEE, 2009.

[7] Halim Alwi and Christopher Edwards. Application of fault tolerant control using sliding modes with on-line control allocation on a large civil aircraft. In *IEEE International Conference on Control Applications. CCA 2007*, pages 1215–1220. IEEE, 2007.

[8] Halim Alwi and Christopher Edwards. Fault detection and fault-tolerant control of a civil aircraft using a sliding-mode-based scheme. *IEEE Transactions on Control Systems Technology*, 16(3):499–510, 2008.

[9] Uri M. Ascher, Robert M.M. Mattheij, and Robert D. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, volume 13 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104, reprint edition, 1994.

[10] Yaakov Bar-Shalom, X. Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. John Wiley & Sons, New York, NY, USA, 2004.

[11] Francois Bateman, Hassan Noura, and Mustapha Ouladsine. Actuators fault diagnosis and tolerant control for an unmanned aerial vehicle. In *Control Applications. CCA 2007. IEEE International Conference on*, pages 1061–1066. IEEE, 2007.

[12] François Bateman, Hassan Noura, and Mustapha Ouladsine. A fault tolerant control strategy for an unmanned aerial vehicle based on a sequential quadratic programming algorithm. In *Decision and Control. CDC 2008. 47th IEEE Conference on*, pages 423–428. IEEE, 2008.

[13] Fares Beainy, Anh Mai, and Sesh Commuri. Unmanned aerial vehicles operational requirements and fault-tolerant robust control in level flight. In *Control and Automation. MED'09. 17th Mediterranean Conference on*, pages 700–705. IEEE, 2009.

[14] Abderrazak I. Belkharraz and Kenneth Sobel. Fault tolerant flight control for a class of control surface failures. In *Proceedings of the 2000 American Control Conference*, volume 6, pages 4209–4213. IEEE, 2000.

[15] Hans Georg Bock and Karl-Josef Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the 1984 IFAC World Congress*, number 9, pages 242–247, 1984.

[16] J.D. Boskovic, Nathan Knoebel, and Raman K. Mehra. An initial study of a combined robust and adaptive control approach to guaranteed performance flight control. In *Proceedings of the 2008 American Control Conference*, pages 5150–5155. IEEE, 2008.

[17] J.D. Boskovic, Sai-Ming Li, and Raman K. Mehra. A decentralized fault-tolerant scheme for flight control applications. In *Proceedings of the 2000 American Control Conference*, volume 6, pages 4214–4218. IEEE, 2000.

[18] J.D. Boskovic, Sai-Ming Li, and R.K. Mehra. On-line failure detection and identification (FDI) and adaptive reconfigurable control (ARC) in aerospace applications. In *Proceedings of the 2001 American Control Conference*, volume 4, pages 2625–2626, 2001.

[19] J.D. Boskovic, Joshua Redding, and Raman K. Mehra. Stable adaptive reconfigurable flight control with self-diagnostics. In *Proceedings of the 2007 American Control Conference. ACC'07*, pages 5765–5770. IEEE, 2007.

[20] Jovan D. Boskovic, Sai-Ming Li, and Raman K. Mehra. On-line failure detection and identification (FDI) and adaptive reconfigurable control (ARC) in aerospace applications. In *Proceedings of the 2001 American Control Conference*, volume 4, pages 2625–2626. IEEE, 2001.

[21] Jovan D. Boskovic and Raman K. Mehra. A multiple model-based reconfigurable flight control system design. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 4, pages 4503–4508. IEEE, 1998.

[22] Jovan D. Boskovic and Raman K. Mehra. Multi-mode switching in flight control. In *Proceedings The 19th Digital Avionics Systems Conference (DASC)*, volume 2, pages 6F2/1–6F2/8. IEEE, 2000.

[23] Jovan D. Boskovic and Raman K. Mehra. Fault accommodation using model predictive methods. In *Proceedings of the 2002 American Control Conference*, volume 6, pages 5104–5109. IEEE, 2002.

[24] Jovan D. Boskovic and Raman K. Mehra. A decentralized fault-tolerant control system for accommodation of failures in higher-order flight control actuators. *Control Systems Technology, IEEE Transactions on*, 18(5):1103–1115, 2010.

[25] Arthur Earl Bryson. *Dynamic Optimization*. Addison Wesley Longman, Menlo Park, CA, 1999.

[26] Eduardo F Camacho, Teodoro Alamo, and David Muñoz de la Pena. Fault-tolerant model predictive control. In *Emerging Technologies and Factory Automation ETFA, 2010 IEEE Conference on*, pages 1–8. IEEE, 2010.

[27] Mark Cannon. Efficient nonlinear model predictive control algorithms. *Annual Reviews in Control*, 28(2):229 – 237, 2004.

[28] J. Chen, R.J. Patton, and Z. Chen. An LMI approach to fault-tolerant control of uncertain systems. In *Proceedings of the 1998 Intelligent Systems and Semiotics (ISAS) and Intelligent Control (ISIC). Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 175–180. IEEE, 1998.

[29] Jerome Cieslak, David Henry, and Ali Zolghadri. Fault tolerant flight control: from theory to piloted flight simulator experiments. *IET Control Theory & Applications*, 4(8):1451–1464, 2010.

[30] Bogdan Ciubotaru and Marcel Staroswiecki. Fault tolerant control of the B747 short-period mode using progressive accommodation. In *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*, pages 3288–3294. IEEE, 2006.

[31] Bruce T. Clough. Unmanned aerial vehicles: Autonomous control challenges, a researcher's perspective. *Journal of Aerospace Computing, Information, and Communication*, 2(8):327–347, 2005.

[32] Robert P. Copeland and Kuldip S. Rattan. A fuzzy logic supervisor for reconfigurable flight control systems. In *Proceedings of the 1994 IEEE National Aerospace and Electronics Conference (NAECON)*, pages 579–586. IEEE, 1994.

[33] Lennon R. Cork, Rodney A. Walker, and Shane Dunn. Fault detection, identification and accommodation techniques for unmanned airborne vehicles. In *Australian International Aerospace Congress (AIAC)*. `http://eprints.qut.edu.au/1729/`, [Last Accessed: 07-01-2016].

[34] Federico Corraro, Adolfo Sollazzo, and Gianfranco Morani. *Fault-Tolerance of a Transport Aircraft with Adaptive Control and Optimal Command Allocation*. INTECH Open Access Publisher, 2011.

[35] Lei Cui, Yong Zhang, and Ying Yang. Youla parameterization based adaptive controller design for fault tolerant control. *Asian Journal of Control*, 16(1):292–295, 2014.

[36] Fabio A. de Almeida and Dirk Leißling. Fault-tolerant model predictive control with flight test results on ATTAS. In *AIAA Guidance, Navigation and Control Conference*, volume 5621, pages 1–23, 2009.

[37] M. Diehl, H.G. Bock, H. Diedam, and P.-B. Wieber. Fast direct multiple shooting algorithms for optimal robot control. In Moritz Diehl and Katja Mombaur, editors, *Fast Motions in Biomechanics and Robotics*, volume 340 of *Lecture Notes in Control and Information Sciences*, pages 65–93. Springer Berlin Heidelberg, 2006.

[38] Moritz Diehl, H. Georg Bock, Johannes P. Schlder, Rolf Findeisen, Zoltan Nagy, and Frank Allgwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[39] Guillaume J.J. Ducard. *Fault-Tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles*. Springer, 2009.

[40] Peter Dyer and Stephen R. McReynolds. *The Computation and Theory of Optimal Control*, volume 177. Academic Press. New York, 1970.

[41] Christopher Edwards, Halim Alwi, and Chee Pin Tan. Sliding mode methods for fault detection and fault tolerant control. In *Control and Fault-Tolerant Systems (SysTol), 2010 Conference on*, pages 106–117. IEEE, 2010.

[42] Christopher Edwards, Thomas Lombaerts, and Hafid Smaili, editors. *Fault Tolerant Flight Control: A Benchmark Challenge*, volume 399 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag Berlin Heidelberg, 2010.

[43] Gamal Elnagar, Mohammad A. Kazemi, and Mohsen Razzaghi. The pseudospectral Legendre method for discretizing optimal control problems. *IEEE Transactions on Automatic Control*, 40(10):1793–1796, 1995.

[44] John S. Eterno, Jerold L. Weiss, Douglas P. Looze, and Alan Willsky. Design issues for fault tolerant-restructurable aircraft control. In *Decision and Control, 1985 24th IEEE Conference on*, volume 24, pages 900–905. IEEE, 1985.

[45] Fariba Fahroo and I. Michael Ross. Direct trajectory optimization by a Chebyshev pseudospectral method. *Journal of Guidance, Control, and Dynamics*, 25(1):160–166, 2002.

[46] J.H. Fan, Y.M. Zhang, and Z.Q. Zheng. Hybrid fault-tolerant flight control against actuator faults and input saturation: a set-invariance approach. In *Proceedings of the 2012 AIAA Guidance, Navigation, and Control Conference, Minneapolis, M.N.*, pages 1–18, 2012. `http://dx.doi.org/10.2514/6.2012-4537`, [Last Accessed: 20-01-2016].

[47] A. Fekih and S.R. Seelem. A fault tolerant control design for automatic steering control of ground vehicles. In *2012 IEEE International Conference on Control Applications (CCA)*, pages 1491–1496, 2012.

[48] Afef Fekih. A robust fault tolerant control strategy for aircraft systems. In *Control Applications, (CCA) & Intelligent Control, (ISIC), IEEE Internation Conference on*, pages 1643–1648. IEEE, 2009.

[49] Afef Fekih and Prasad Pilla. A passive fault tolerant control strategy for the uncertain MIMO aircraft model F-18. In *System Theory. SSST'07. Thirty-Ninth Southeastern Symposium on*, pages 320–325. IEEE, 2007.

[50] S. Ganguli, A. Marcos, and Gary Balas. Reconfigurable LPV control design for Boeing 747-100/200 longitudinal axis. In *Proceedings of the 2002 American Control Conference*, volume 5, pages 3612–3617 vol.5, 2002.

[51] Gang Gao and Jinzhi Wang. Observer-based fault-tolerant control for an air-breathing hypersonic vehicle model. *Nonlinear Dynamics*, 76(1):409–430, 2014.

[52] Frederico R. Garza and Eugene A. Morelli. A collection of nonlinear aircraft simulations in Matlab. *NASA Langley Research Center, Technical Report NASA/TM*, 212145, 2003.

[53] Shreekant Gayaka and Bin Yao. Accommodation of partial actuator faults using output feedback based adaptive robust control. In *Prognostics and Health Management. PHM 2008. International Conference on*, pages 1–9. IEEE, 2008.

[54] Philip E. Gill, Walter Murray, and Michael A. Saunders. Users guide for SNOPT version 7: Software for large-scale nonlinear programming. 2006. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.217.5929`, [Last Accessed: 20-01-2016].

[55] M. Gopinathan, J.D. Boskovic, R.K. Mehra, and C. Rago. A multiple model predictive scheme for fault-tolerant flight control design. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 2, pages 1376–1381, 1998.

[56] Murali Gopinathan, Jovan D. Boskovic, Raman K. Mehra, and Constantino Rago. A multiple model predictive scheme for fault-tolerant flight control design. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 2, pages 1376–1381. IEEE, 1998.

[57] David Gottlieb and Eitan Tadmor. Recovering pointwise values of discontinuous data within spectral accuracy. In *Progress and Supercomputing in Computational Fluid Dynamics*, pages 357–375. Springer, 1985.

[58] Leslie Greengard. Spectral integration and two-point boundary value problems. *SIAM Journal on Numerical Analysis*, 28(4):1071–1080, 1991.

[59] Harry N. Gross, Phillip R. Chandler, and Robert A. Eslinger. Renewed interest in hinge moment models for failure detection and isolation. In *Proceedings of the 1986 American Control Conference*, pages 1497–1502. IEEE, 1986.

[60] Yu Guan, Yingchun Zhang, and Xueqin Chen. Application of robust fault-tolerant control in satellite attitude control system. In *2010 3rd International Symposium on Systems and Control in Aeronautics and Astronautics (ISSCAA)*, pages 23–28, 2010.

[61] Yuying Guo, Youmin Zhang, and Bin Jiang. Multi-model-based flight control system reconfiguration control in the presence of input constraints. In *Intelligent Control and Automation (WCICA), 8th World Congress on*, pages 5819–5824, 2010.

[62] Ralf Hannemann-Tams and Wolfgang Marquardt. How to verify optimal controls computed by direct shooting methods? a tutorial. *Journal of Process Control*, 22(2):494 – 507, 2012.

[63] G. Hein and M.D.F. Bento. Unmanned aerial vehicles: An overview. *InsideGNSS*, 3(1):54–61, 2008.

[64] Michael A. Henson. Nonlinear model predictive control: current status and future directions. *Computers and Chemical Engineering*, 23(2):187–202, 1998.

[65] Moshe Idan, Matthew Johnson, Anthony J. Calise, and John Kaneshige. Intelligent aerodynamic/propulsion flight control for flight safety: a nonlinear adaptive approach. In *Proceedings of the 2001 American Control Conference*, volume 4, pages 2918–2923. IEEE, 2001.

[66] The MathWorks Inc. Matlab release 2014a aerospace toolbox, 2014.

[67] Jin Jiang. Design of reconfigurable control systems using eigenstructure assignments. *International Journal of Control*, 59(2):395–410, 1994.

[68] Jin Jiang and Xiang Yu. Fault-tolerant control systems: A comparative study between active and passive approaches. *Annual Reviews in Control*, 36(1):60–72, 2012.

[69] Damien B. Jourdan, Michael D. Piedmonte, Vlad Gavrilets, David W. Vos, and Jim McCormick. Enhancing UAV survivability through damage tolerant control. In *AIAA Guidance, Navigation, and Control Conference*, pages 2–5, 2010.

[70] M. M. Kale and A.J. Chipperfield. Robust and stabilized MPC formulations for fault tolerant and reconfigurable flight control. In *Proceedings of the 2004 IEEE International Symposium on Intelligent Control*, pages 222–227, 2004.

[71] M.S. Keating, M. Pachter, and C.H. Houpis. QFT applied to fault tolerant flight control system design. In *Proceedings of the 1995 American Control Conference*, volume 1, pages 184–188. IEEE, 1995.

[72] Rudaba Khan, Paul Williams, Robin Hill, and Cees Bil. Fault tolerant flight control system design for UAV using nonlinear model predictive control. In *Procceedings of 2011 Australian Control Conference (AUCC)*, pages 297–302. Barton, A.C.T.: Engineers Australia, November 2011. `http://search.informit.com.au/documentSummary;dn=211082540617900;res=IELENG`.

[73] Rudaba Khan, Paul Williams, Robin Hill, and Cees Bil. Fault tolerant flight control system design for unmanned air vehicles. In *Procceedings of Third Asia-Pacific International Symposium on Aerospace Technology (APISAT 2011)*, pages 260–268. Barton, A.C.T.: Engineers Australia, March 2011. `http://search.informit.com.au/documentSummary;dn=368247541391538;res=IELENG`.

[74] David Kincaid and E.W. Cheney. *Numerical Analysis: Mathematics of Scientific Computing*, volume 2 of *Pure and Applied Undergraduate Texts*. American Mathematical Society, third edition, 2002.

[75] C. Kirches, L. Wirsching, H.G. Bock, and J.P. Schlder. Efficient direct multiple shooting for nonlinear model predictive control on long horizons. *Journal of Process Control*, 22(3):540 – 550, 2012.

[76] Thomas Krüger, Philipp Schnetter, Robin Placzek, and Peter Vörsmann. Fault-tolerant nonlinear adaptive flight control using sliding mode online learning. *Neural Networks*, 32:267–274, 2012.

[77] Soonman Kwon, Jong-Min Cheon, Jongmoo Lee, Choon-Kyung Kim, and Seog-Joo Kim. A design and implementation of a fault-tolerant rod control system for nuclear power plants. In *2006 IEEE International Symposium on Industrial Electronics*, volume 3, pages 1933–1936, 2006.

[78] Yan Li, Zhifeng Wang, N. Sundararajan, and P. Saratchandran. Fault tolerant neuro-H∞; controller design for aircraft auto-landing. In *Proceedings of the 35th Southeastern Symposium on System Theory*, pages 172–175, 2003.

[79] Fang Liao, Jian Liang Wang, and Guang-Hong Yang. Reliable robust flight tracking control: an LMI approach. *IEEE Transactions on Control Systems Technology*, 10(1):76–89, 2002.

[80] George Limnaios. Current usage of unmanned aircraft systems (UAS) and future challenges: A mission oriented simulator for UAS as a tool for design and performance evaluation. *Journal of Computations and Modelling*, 4(1):167–188, 2014.

[81] Chun-Liang Lin and Chun-Te Liu. Failure detection and adaptive compensation for fault tolerable flight control systems. *IEEE Transactions on Industrial Informatics*, 3(4):322–331, 2007.

[82] Jianwei Liu, Bin Jiang, and Ke Zhang. Fault diagnosis for linear discrete systems based on an adaptive observer. Journal of Mathematical Problems in Engineering, Hindawi Publishing Corporation. Vol. 2013, Article ID 343524, 5 pages, 2013. `http://dx.doi.org/10.1155/2013/343524`, [Last Accessed: 07-01-2016].

[83] Xiaoxiong Liu, Yan Wu, Jingping Shi, and Weiguo Zhang. Adaptive fault-tolerant flight control system design using neural networks. In *IEEE International Conference on Industrial Technology, ICIT 2008*, pages 1–5. IEEE, 2008.

[84] Thomas Lombaerts, Herve Huisman, Ping Chu, Jan A. Mulder, and Diederick Joosten. Nonlinear reconfiguring flight control based on online physical model identification. *Journal of Guidance, Control, and Dynamics*, 32(3):727–748, 2009.

[85] C.E. Long, P.K. Polisetty, and E.P. Gatzke. Nonlinear model predictive control using deterministic global optimization. *Journal of Process Control*, 16(6):635 – 643, 2006.

[86] Jan M. Maciejowski. The implicit daisy-chaining property of constrained predictive control. *Applied Mathematics and Computer Science*, 8:695–712, 1998.

[87] Jan M. Maciejowski and Colin N. Jones. MPC fault-tolerant flight control case study: Flight 1862. In *IFAC Safeprocess Conference*, pages 121–126, 2003.

[88] J.M. Maciejowski. Fault-tolerant aspects of MPC. In *IEE Two-Day Workshop on Model Predictive Control: Techniques and Applications-Day 2 (Ref. No. 1999/096)*, pages 1/1–1/4. IET, 1999.

[89] J.M. Maciejowski. Modelling and predictive control: Enabling technologies for reconfiguration. *Annual Reviews in Control*, 23:13–23, 1999.

[90] J.M. Maciejowski. *Predictive Control: With Constraints*. Pearson Education. Prentice Hall, 2002.

[91] W.D. Morse and A. Ossman. Model following reconfigurable flight control system for the AFTI/F-16. *Journal of Guidance, Control, and Dynamics*, 13(6):969–976, 1990.

[92] Marcello R. Napolitano, Younghwan An, and Brad A. Seanor. A fault tolerant flight control system for sensor and actuator failures using neural networks. *Aircraft Design*, 3(2):103 – 128, 2000.

[93] M. Pachter and C.H. Houpis. A full envelope flight control system design, including aerodynamic control effector failures accommodation, using QFT. In *Proceedings of the 1997 Symposium on QFT and other Frequency Domain Methods and Applications*, pages 45–54. DTIC Document, 1997.

[94] Meir Pachter and Yih-Shiun Huang. Fault tolerant flight control. *Journal of Guidance, Control, and Dynamics*, 26(1):151–160, 2003.

[95] A.A. Pashilkar, N. Sundararajan, and P. Saratchandran. A fault-tolerant neural aided controller for aircraft auto-landing. *Aerospace Science and Technology*, 10(1):49–61, 2006.

[96] R.J. Patton. Fault detection and diagnosis in aerospace systems using analytical redundancy. *Computing & Control Engineering Journal*, 2(3):127–136, 1991.

[97] Ron J. Patton. Fault-tolerant control systems: The 1997 situation. In *IFAC Symposium on Fault Detection Supervision and Safety for Technical Processes*, volume 3, pages 1033–1054, 1997.

[98] Li Peng, Ma Jian-jun, Li Wen-qiang, and Zheng Zhi-qiang. Adaptive conditional integral sliding mode control for fault tolerant flight control system. In *Asia Simulation Conference-*

7th International Conference on System Simulation and Scientific Computing, ICSC 2008, pages 638–642. IEEE, 2008.

[99] Mario G. Perhinschi, J. Burken, and G. Campa. Comparison of different neural augmentations for the fault tolerant control laws of the WVU YF-22 model aircraft. In *14th Mediterranean Conference on Control and Automation, 2006. MED'06*, pages 1–6. IEEE, 2006.

[100] Mario G. Perhinschi, Hever Moncayo, and Jennifer Davis. Integrated framework for artificial immunity-based aircraft failure detection, identification, and evaluation. *Journal of Aircraft*, 47(6):1847–1859, 2010.

[101] M.G. Perhinschi, J. Burken, M.R. Napolitano, G. Campa, and M.L. Fravolini. Performance comparison of different neural augmentation for the NASA Gen-2 IFCS F-15 control laws. In *Proceedings of the 2004 American Control Conference*, volume 4, pages 3180–3184. IEEE, 2004.

[102] Marios Polycarpou, Xiaodong Zhang, Roger Xu, Yanli Yang, and Chiman Kwan. A neural network based approach to adaptive fault tolerant flight control. In *Proceedings of the 2004 IEEE International Symposium on Intelligent Control*, pages 61–66. IEEE, 2004.

[103] Ian Postlethwaite, Emmanuel Prempain, Ercument Turkoglu, Matthew C. Turner, Kris Ellis, and A.W. Gubbels. Design and flight testing of various H∞ controllers for the Bell 205 helicopter. *Control Engineering Practice*, 13(3):383–398, 2005.

[104] S. Joe Qin and Thomas A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[105] Martin Rau and Dierk Schroder. Model predictive control with nonlinear state space models. In *7th International Workshop on Advanced Motion Control*, pages 136–141. IEEE, 2002.

[106] Branko Ristic, Sanjeev Arulampalm, and Neil Gordon. *Beyond the Kalman filter : particle filters for tracking applications*. Artech House Boston, Ma. ; London, 2004.

[107] I. Michael Ross and Fariba Fahroo. Legendre pseudospectral approximations of optimal control problems. In *New Trends in Nonlinear Dynamics and Control and their Applications*, pages 327–342. Springer, 2003.

[108] I. Michael Ross and Mark Karpenko. A review of pseudospectral optimal control: From theory to flight. *Annual Reviews in Control*, 36(2):182–197, 2012.

[109] T. Sadeghi and K. Lai. Integrated fault tolerant control law. In *Proceedings of the IEEE 1995 National Aerospace and Electronics Conference (NAECON)*, volume 1, pages 491–502. IEEE, 1995.

[110] Montadher Sami and Ron J. Patton. Active fault tolerant control for nonlinear systems with simultaneous actuator and sensor faults. *International Journal of Control, Automation and Systems*, 11(6):1149–1161, 2013.

[111] Andreas Schfer, Peter Khl, Moritz Diehl, Johannes Schlder, and Hans Georg Bock. Fast reduced multiple shooting methods for nonlinear model predictive control. *Chemical Engineering and Processing: Process Intensification*, 46(11):1200 – 1214, 2007. Special Issue on Process Optimization and Control in Chemical Engineering and Processing.

[112] Jong-Yeob Shin and Irene Gregory. Robust gain-scheduled fault tolerant control for a transport aircraft. In *IEEE International Conference on Control Applications, 2007. CCA 2007*, pages 1209–1214. IEEE, 2007.

[113] C. Sloth, T. Esbensen, and J. Stoustrup. Active and passive fault-tolerant LPV control of wind turbines. In *Proceedings of the 2010 American Control Conference (ACC)*, pages 4640–4646, 2010.

[114] Robert F. Stengel. *Flight Dynamics*. Princeton University Press, 2015.

[115] Brian L. Stevens and Frank L. Lewis. *Aircraft Control and Simulation*. John Wiley & Sons, 2003.

[116] Jasem Tamimi and Pu Li. A combined approach to nonlinear model predictive control of fast systems. *Journal of Process Control*, 20(9):1092 – 1102, 2010. ADCHEM 2009 Special Issue.

[117] Yimeng Tang and R.J. Patton. Reconfigurable flight control using feedback linearization with online neural network adaption. In *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, pages 324–329, Oct 2013.

[118] Amir Torabi, Sobhan Salehi, Ali Karsaz, and Ebrahim Tarsayi. Predictive controller for pitch controller aircraft. *Journal of American Science*, 10(1):193–198, 2014.

[119] David W. Vos and Ben Motazed. Application of fault-tolerant controls to UAVs. In *Aerospace/Defense Sensing and Controls*, pages 69–75. International Society for Optics and Photonics, 1996.

[120] E.A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *The IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pages 153–158, 2000.

[121] Wei Wang, T Hameed, and Zhang Ren. Extended state observer-based robust fault-tolerant controller for flight control surface failures. In *ICEMI'09. 9th International Conference on Electronic Measurement & Instruments*, pages 3–610. IEEE, 2009.

[122] Wei Wang, Tahir Hameed, Kefeng Li, and Zhang Ren. A new approach for fault tolerance/disturbance rejection control. In *WCICA 2008. 7th World Congress on Intelligent Control and Automation, 2008*, pages 1405–1408. IEEE, 2008.

[123] Paul Williams. Jacobi pseudospectral method for solving optimal control problems. *Journal of Guidance, Control, and Dynamics*, 27(2):293–297, 2004.

[124] Paul Williams. A comparison of differentiation and integration based direct transcription methods (AAS 05-128). *Advances in the Astronautical Sciences*, 120(1):389, 2005.

[125] Paul Williams. Hermite-Legendre-Gauss-Lobatto direct transcription in trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 32(4):1392–1395, 2009.

[126] Paul Williams. Quadrature discretization method in tethered satellite control. *Applied Mathematics and Computation*, 217(21):8223–8235, 2011.

[127] Paul Williams. Outer loop guidance model. Private Communication, 2012.

[128] Paul Williams. UAV model. Private Communication, 2015.

[129] N Eva Wu and Tijian Chen. Failure sensitivity and robustness in reconfigurable flight control systems. In *Proceedings of the IEEE 1994 National Aerospace and Electronics Conference*, pages 548–555. IEEE, 1994.

[130] Shu-Fan Wu, Michael J Grimble, and Wei Wei. QFT based robust/fault tolerant flight control design for a remote pilotless vehicle. In *Proceedings of the 1999 IEEE International Conference on Control Applications*, volume 1, pages 57–62. IEEE, 1999.

[131] Qi Xiao-hui and Zhen Hong-tao. The design of adaptive sliding mode control law on reconfigurable flight control system. In *2011 International Conference on Electronics, Communications and Control (ICECC)*, pages 332–536. IEEE, 2011.

[132] Jiang Xiaosong. An utility scheme for online fault detection. In *International Forum on Information Technology and Applications, 2009*, volume 3, pages 264–267, 2009.

[133] Li Yan, N. Sundararajan, and P. Saratchandran. Fault tolerant flight controller using minimal resource allocating neural networks MRAN. In *Proceedings of the 1999 American Control Conference*, volume 4, pages 2605–2609, 1999.

[134] Guang-Hong Yang and Kai-Yew Lum. Fault-tolerant flight tracking control with stuck faults. In *Proceedings of the 2003 American Control Conference*, volume 1, pages 521–526. IEEE, 2003.

[135] Zhi-Jun Yang, Xiao-Hui Qi, and Gan-Lin Shan. Adaptive sliding-mode tracking control laws applied to fault-tolerant flight control systems. In *ISA 2009. International Workshop on Intelligent Systems and Applications, 2009*, pages 1–4. IEEE, 2009.

[136] Zhi-jun Yang, Xiao-hui Qi, and Gan-lin Shan. Fault tolerant flight control law design with a model-following sliding mode controller. In *Proceedings of the 2009 International Conference on Measuring Technology and Mechatronics Automation*, pages 665–668. IEEE Computer Society, 2009.

[137] Dan Ye and Guang-Hong Yang. Adaptive fault-tolerant tracking control against actuator faults with application to flight control. *IEEE Transactions on Control Systems Technology*, 14(6):1088–1096, 2006.

[138] Sijun Ye, Youmin Zhang, C. Rabbath, Xinmin Wang, and Yan Li. An LMI approach to mixed H1/H∞; robust fault-tolerant control design with uncertainties. In *Proceedings of the 2009 American Control Conference*, pages 5540–5545, 2009.

[139] Sijun Ye, Youmin Zhang, Xinmin Wang, and Bin Jiang. Fault-tolerant control for a class of uncertain systems with actuator faults. *Tsinghua Science and Technology*, 15(2):174–183, 2010.

[140] Sijun Ye, Youmin Zhang, Xinmin Wang, and C.-A. Rabbath. Robust fault-tolerant control using on-line control re-allocation with application to aircraft. In *Proceedings of the 2009 American Control Conference*, pages 5534–5539, 2009.

[141] Jiong-Sang Yee, Bin Jiang, Jian Liang Wang, and Yan Li. Discrete-time actuator fault estimation design for flight control application. In *7th International Conference on Control, Automation, Robotics and Vision*, volume 3, pages 1287–1292, 2002.

[142] Bin Yu, Youmin Zhang, Jianguo Yan, Yaohong Qu, and Zhuang Liu. Fault tolerant control using linear quadratic technique against actuator faults in a UAV. In *32nd Chinese Control Conference (CCC)*, pages 6294–6299. IEEE, 2013.

[143] Xiang Yu and Jin Jiang. Fault-tolerant flight control system design against control surface impairments. *IEEE Transactions on Aerospace and Electronic Systems*, 48(2):1031–1051, 2012.

[144] Xiang Yu and Jin Jiang. Hybrid fault-tolerant flight control system design against partial actuator failures. *IEEE Transactions on Control Systems Technology*, 20(4):871–886, 2012.

[145] Xiang Yu, Yan Li, Xinmin Wang, and Kairui Zhao. An autonomous robust fault tolerant control system. In *2006 IEEE International Conference on Information Acquisition*, pages 1191–1196, 2006.

[146] Xiaodong Zhang, M.M. Polycarpou, R. Xu, and Chiman Kwan. Actuator fault diagnosis and accommodation for flight safety. In *Proceedings of the 2005 IEEE International Symposium on Intelligent Control, Mediterrean Conference on Control and Automation*, pages 640–645, 2005.

[147] Youmin Zhang and Jin Jiang. Fault tolerant control system design with explicit consideration of performance degradation. *IEEE Transactions on Aerospace and Electronic Systems*, 39(3):838–848, 2003.

[148] Youmin Zhang and Jin Jiang. Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control*, 32(2):229–252, 2008.

[149] D. Zumoffen, M. Basualdo, M. Jordan, and A. Ceccatto. Robust adaptive predictive fault-tolerant control linked with fault diagnosis system applied on a nonlinear chemical process. In *2006 45th IEEE Conference on Decision and Control*, pages 3512–3517, 2006.